Basisskript zur Vorlesung

Einführung in die Mathematik des Operations Research

Sommersemester 2019

Dr. Sven Mallach

Department Mathematik und Informatik
Universität zu Köln
Köln, Deutschland

Dieses Basisskript basiert weitgehend auf und erweitert die Vorarbeiten von
Herrn Prof. Dr. Frank Vallentin,
sowie
Frau Dr. Anna Gundert
und
Herrn Dr. Frederik von Heymann,
denen an dieser Stelle ausdrücklich mein Dank gebührt.

Teil A

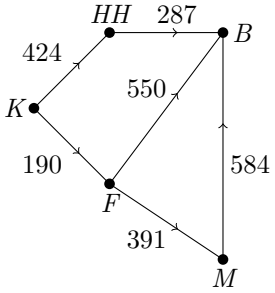
Graphen und Netzwerke, kombinatorische Algorithmen

KAPITEL I

Kürzeste Wege

Das Ziel dieses Kapitels ist es, kürzeste Wege in einem vorgegebenen Netzwerk zu verstehen und zu berechnen.
Ein typisches Anwendungsgebiet der Bestimmung kürzester Wege in einem Netzwerk sind Navigationsgeräte.

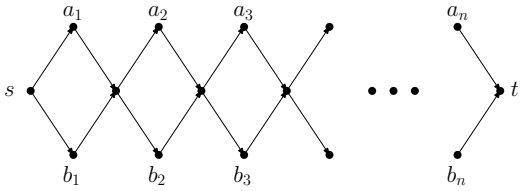
BEISPIEL 0.1. *Als einführendes Beispiel für ein Netzwerk betrachten wir das (sehr stark vereinfachte) Straßennetz(werk) zwischen den fünf Städten Köln (K), Hamburg (HH), München (M), Frankfurt (F) und Berlin (B):*



Die Zahlen geben hier die Entfernungen zwischen den Städten an. Ein Ziel könnte zum Beispiel sein, den kürzesten Weg von Köln nach Berlin zu finden, welcher in diesem Netzwerk auch leicht zu erkennen ist, wenn wir alle möglichen Wege von Köln nach Bonn aufzählen und uns dann den kürzesten auswählen.
$K \to F \to M \to B$: 1165
$K \to F \to B$: 740
$K \to HH \to B$: 711

In größeren Netzwerken sind kürzeste Wege jedoch nicht so einfach zu finden. Man kann dann in der Praxis nicht mehr alle Möglichkeiten ausprobieren, weil es davon im Allgemeinen zu viele gibt, um dies in angemessener Zeit zu schaffen.

BEISPIEL 0.2. *In folgendem Graphen haben wir $3n+1$ Knoten, jedoch eine Anzahl von Wegen zwischen s und t, die exponentiell in n ist.*



Ungefähre Rechenzeit (bei benötigten 10^{-9} Sekunden pro Auswertung eines Wegs):

n	*Rechenzeit*
20	$0,001s$
40	18 *min*
60	36 *J.*
80	38 *Mio. J.*

Bevor wir uns mit mathematischen Methoden zur Bestimmung kürzester Wege beschäftigen können, müssen wir uns zunächst mit einigen Grundbegriffen vertraut machen. Wir müssen vor allem definieren, was wir unter einem kürzesten Weg verstehen.

1. Gerichtete Graphen

DEFINITION 1.1. *Ein gerichteter Graph (Netzwerk) $D = (V, A)$ ist ein Paar, bestehend aus einer endlichen Menge V, der Menge der* Knoten *von D, und aus einer endlichen Menge*

$$A \subseteq \{(v, w) \in V \times V : v \neq w\},$$

der Menge der gerichteten Kanten *von D.*

Mit anderen Worten: Ein gerichteter Graph beschreibt eine irreflexive Relation auf $V \times V$.

DEFINITION 1.2. *Sei $D = (V, A)$ ein gerichteter Graph.*

(1) *Eine* Kantenfolge *P in D ist ein Tupel der Form*

$$P = (v_0, a_1, v_1, a_2, v_2, \ldots, a_m, v_m)$$

mit den Eigenschaften $v_0, \ldots, v_m \in V$, $a_1, \ldots, a_m \in A$ und $a_i = (v_{i-1}, v_i)$ für $i = 1, \ldots, m$. Der Knoten v_0 heißt Startknoten *von P und der Knoten v_m heißt* Endknoten *von P. Wir sprechen auch von einer v_0-v_m-Kantenfolge.*

(2) *Eine Kantenfolge P heißt* einfacher v_0-v_m-Weg *oder kurz* Weg*, falls*

$$|\{v_0, \ldots, v_m\}| = m + 1$$

gilt, d.h. die in P auftretenden Knoten sind alle paarweise verschieden.

DEFINITION 1.3. *Sei $D = (V, A)$ ein gerichteter Graph. Eine Kantenfolge P in D heißt (einfacher, gerichteter)* Kreis*, falls $v_0 = v_m$, und*

$$|\{v_0, \ldots, v_m\}| = m$$

gilt.

DEFINITION 1.4. *Sei $D = (V, A)$ ein gerichteter Graph.*

(1) *Eine Funktion*

$$\ell \colon A \to \mathbb{R},$$

heißt Kantenlängenfunktion.

(2) *Mit*

$$\ell(P) = \sum_{i=1}^{m} \ell(a_i).$$

bezeichnen wir in Kurzschreibweise die Länge *einer Kantenfolge P.*

(3) *Für zwei Knoten s und t in V definieren wir*

$$\operatorname{dist}(s, t) = \inf\{\ell(P) : P \text{ ist ein (einfacher) } s\text{-}t\text{-Weg}\}.$$

als den Abstand (die Distanz) *von t ausgehend von s.*

(4) *Ist eine Kantenfolge P ein (einfacher) s-t-Weg mit $\ell(P) = \operatorname{dist}(s, t)$, dann heißt P ein kürzester s-t-Weg.*

BEMERKUNG 1.5.

(1) *Es kann durchaus vorkommen, dass Kanten eine negative Länge besitzen. Modernes Beispiel: Modellierung von Batterieverbrauch über Fahrtstrecken, und Verwendung von „Aufladekanten“ um entsprechende Stationen in das modellierte Netzwerk zu integrieren.*

(2) *Manchmal existiert kein Weg von s nach t und somit ist* $\operatorname{dist}(s, t) = \infty$*. Dies erklärt, warum wir in der Definition des Abstandes ein Infimum und kein Maximum angegeben haben.*

(3) *Wenn es einen Weg von s nach t gibt, dann auch einen kürzesten. Dieser ist jedoch nicht notwendigerweise eindeutig.*
(4) *Die Distanz eines Knoten zu sich selbst ist, per Definition, immer gleich Null:* $\operatorname{dist}(s,s) = 0$*, weil* $P = (s)$ *der eindeutige Weg von s nach s ist, und dieser besitzt Länge Null.*
(5) *Da ein (einfacher) Weg* $P = (v_0, a_1, v_1, \ldots, a_m, v_m)$ *durch die in ihm enthaltenen Kanten eindeutig definiert ist, fassen wir P im Folgenden oft vereinfachend als Kantenmenge (bzw. -folge) auf und schreiben dann* $P = \{a_1, \ldots, a_m\} \subseteq A$.

BEMERKUNG 1.6. *Manchmal verwenden wir in Vorlesung und Übung die folgende vereinfachende Notation: Sei $D = (V, A)$ ein gerichteter Graph. Mit $\mathbb{R}^V$ (bzw. $\mathbb{R}^A$) bezeichnen wir den Vektorraum der Funktionen von V (bzw. A) nach $\mathbb{R}$, also*

$$\mathbb{R}^V = \{f \colon V \to \mathbb{R} : f \text{ Funktion}\} \quad \text{und analog}$$
$$\mathbb{R}^A = \{f \colon A \to \mathbb{R} : f \text{ Funktion}\}$$
$$\mathbb{R}^{V \times A} = \{f \colon V \times A \to \mathbb{R} : f \text{ Funktion}\}$$

2. Potentiale

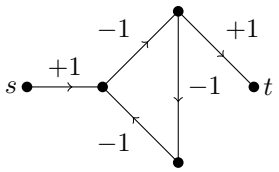
Wie kann man beweisen, dass ein gegebener Weg von s nach t tatsächlich ein kürzester s-t-Weg ist?

Schon gesehen: Alle s-t-Wege aufzuzählen kann sehr ineffizient sein, da es exponentiell viele Wege von s nach t geben kann.

Ziel dieses Unterkapitels: Eine alternative, viel effizientere Beweismethode finden.

Erste Hoffnung: Kürzeste s-t-Kantenfolgen entsprechen stets kürzesten s-t-Wegen. Wir könnten dann das Problem auf das Berechnen ersterer reduzieren, und müssten in einer Beweisführung zur Minimalität einer s-t-Kantenfolge nicht *explizit* gewährleisten, dass in dieser keine *Knoten* doppelt auftreten.
Der verwendete Konjunktiv deutet aber schon daraufhin, dass dies leider nicht so ist, wie folgendes Beispiel zeigt.

BEISPIEL 2.1. *In dem Graph*



ist $dist(s,t) = 1-1+1 = 1$ *(es existiert also ein kürzester s-t-Weg), aber es existiert keine kürzeste s-t-Kantenfolge.*

Das Problem in Beispiel 2.1 ist die Existenz eines gerichteten Kreises, der eine negative Länge besitzt. Da dieser beliebig oft durchlaufen werden kann, existiert keine kürzeste s-t-Kantenfolge. Tatsächlich ist das Problem der Bestimmung eines kürzesten *s-t-Weges* in diesem Fall (vermutlich) nicht effizient lösbar.
Die Behandlung dieser Frage geht jedoch über den Stoff dieser Vorlesung hinaus, wird aber z.B. in „Algorithmen der linearen und diskreten Optimierung“ umrissen[1].

[1]Für Neugierige: Die Fragestellung hängt mit dem *Hamiltonpfadproblem* zusammen und ist äquivalent zur $\mathcal{P} \stackrel{?}{=} \mathcal{NP}$-Frage in der Informatik, deren Beantwortung mit 1 Mio. Dollar dotiert ist.

Wenn man Kreise negativer Länge nun ausschließt, ist es immerhin stets möglich in der Menge aller kürzesten Kantenfolgen von s nach t (sofern diese nicht leer ist) einen Weg zu finden, wie der folgende erste Satz aussagt. Mehr noch ist dieser Weg dann unweigerlich auch ein kürzester Weg von s nach t. Andererseits überlegt man sich leicht, dass nicht jede kürzeste s-t-Kantenfolge ein Weg ist.

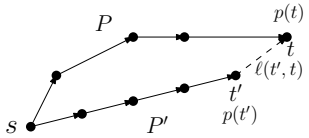
SATZ 2.2. *Es sei $D = (V, A)$ ein gerichteter Graph mit Längenfunktion $\ell : A \to \mathbb{R}$. Angenommen alle gerichteten Kreise in D haben nicht-negative Länge und angenommen es gibt einen s-t-Weg, d.h. $\operatorname{dist}(s,t) < \infty$. Dann existiert eine kürzeste Kantenfolge mit Startknoten s und Endknoten t, die ein Weg ist.*

Zurück zur Ausgangsfrage: Wie beweist man, dass ein gegebener s-t-Weg P ein kürzester ist? Bzw. wie erkennt man, dass ein s-t-Weg P *kein* kürzester ist?

Offensichtlicher, aber ggf. aufwändiger Beweis: Einen weiteren s-t-Weg angeben, der kürzer ist.
Effizienteres (lokales) Kriterium: Konzept der Potentiale:
Angenommen, ein s-t-Weg P hat die Länge $p(t)$ und es gibt einen Knoten t' mit $(t', t) \in A$. Sei zudem P' ein s-t'-Weg der Länge $p(t')$. Falls die Ungleichung

$$p(t) - p(t') \leq \ell((t', t))$$

nicht erfüllt ist, dann kann $p(t)$ nicht die Länge eines kürzesten Weges sein, weil die Kantenfolge, die wir erhalten wenn wir die Kante (t', t) an P' anhängen nur Länge $p(t') + \ell((t', t))$ hat. $\Rightarrow$ Prinzip: Kürzeste Wege bestehen aus kürzesten Teilwegen.



DEFINITION 2.3. *Sei $D = (V, A)$ ein gerichteter Graph mit Kantenlängenfunktion $\ell\colon A \to \mathbb{R}$. Eine Funktion $p\colon V \to \mathbb{R}$ heißt* Potential *für D und ℓ, falls die Ungleichung*

$$p(v) - p(u) \leq \ell(a) \quad \text{für alle Kanten } a = (u, v) \in A$$

gilt.

BEMERKUNG 2.4.

(1) *Offensichtlich ist $p = 0$ ein Potential, falls die Kantenlängenfunktion nur nicht-negative Werte annimmt.*
(2) *Mithilfe eines bekannten Potentials p können wir eine nicht-negative Kantenlängenfunktion ℓ' erzeugen, so dass P ein kürzester s-t-Weg bzgl. ℓ' ist, genau dann wenn P ein kürzester s-t-Weg bzgl. ℓ ist. Setze dazu $\ell'(a) = \ell(a) - p(v) + p(u)$ für alle $(u, v) \in A$.*

BEMERKUNG 2.5. *Eine für das tiefere Verständnis von Potentialen nützliche Erkenntnis ist, dass im obigen Beispiel auch P' kein kürzester s-t'-Weg gewesen sein muss, um P als* nicht-*kürzesten s-t-Weg zu zertifizieren. D.h. auch für $p(t') > \operatorname{dist}(s, t')$ kann $p(t)$ <u>nicht</u> die Länge eines kürzesten s-t-Weges sein, falls $p(t) - p(t') > \ell((t', t))$, p also <u>kein</u> Potential ist.*
Daraus erwächst der Ansatz mit einem p zu starten, das die wahre Distanz von s zu anderen Knoten höchstens überschätzt (also $p(v) \geq \operatorname{dist}(s, v)$ gilt für alle $v \in V$), und dieses p sukzessive zu verbessern, bis es ein Potential ist. Wir werden auf diesen Ansatz später noch einmal zurückkommen.

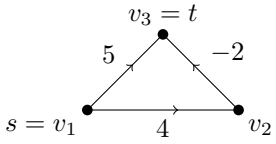
SATZ 2.6. *Sei $D = (V, A)$ ein gerichteter Graph mit Kantenlängenfunktion $\ell\colon A \to \mathbb{R}$. Ein Potential $p\colon V \to \mathbb{R}$ für D und ℓ existiert genau dann, wenn alle gerichteten Kreise in D nicht-negative Länge besitzen.*

SATZ 2.7. *(geometrische Modellierung kürzester Wege mit linearen Ungleichungen) Es sei $D = (V, A)$ ein gerichteter Graph und $\ell\colon A \to \mathbb{R}$ eine Kantenlängenfunktion. Angenommen, alle gerichteten Kreise in D haben nicht-negative Länge. Seien s, $t \in V$ und angenommen, für alle Knoten $v \in V$ gilt $\operatorname{dist}(s, v) < \infty$. Dann gilt die folgende Min-Max-Charakterisierung für die Länge eines kürzesten Weges von s nach t:*

$$\begin{aligned}\operatorname{dist}(s,t) &= \min\{\ell(P) : P \text{ ist } s\text{-}t\text{-Weg}\} \\ &= \max\{p(t) - p(s) : p\colon V \to \mathbb{R},\ p(v) - p(u) \le \ell(a) \text{ für alle } a = (u,v) \in A\}\end{aligned}$$

Bevor wir den Satz beweisen, betrachten wir ein Beispiel.

BEISPIEL 2.8. *Wir wenden den Satz auf den Graphen*



an. Man erhält

$$\begin{aligned} dist(v_1, v_3) &= \max \quad p(v_3) - p(v_1) \\ &\qquad\quad p(v_2) - p(v_1) \le 4 \\ &\qquad\quad p(v_3) - p(v_1) \le 5 \\ &\qquad\quad p(v_3) - p(v_2) \le -2 \\ &\qquad\quad p(v_1), p(v_2), p(v_3) \in \mathbb{R} \\ &= \max \quad (-1,0,1)p \\ &\qquad\quad (-1,1,0)p \le 4 \\ &\qquad\quad (-1,0,1)p \le 5 \\ &\qquad\quad (0,-1,1)p \le -2 \\ &\qquad\quad p \in \mathbb{R}^3 \end{aligned}$$

Dabei können wir das Potential p auch als Spaltenvektor

$$p = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \in \mathbb{R}^3$$

schreiben, wobei wir dem Funktionswert $p(v_i)$ den Eintrag p_i zuordnen. Auf diese Notation wird bei der Modellierung im Operations Research häufig zurückgegriffen. Dadurch wird noch deutlicher, dass durch die drei linearen Ungleichungen, die der Vektor p erfüllen muss, ein geometrisches Gebilde im Raum $\mathbb{R}^3$ beschrieben wird. Der Vektor $p = (0,4,1)^\mathsf{T}$ erfüllt die drei linearen Ungleichungen. Mit Satz 2.7 ergibt dies unmittelbar einen Beweis (ein Zertifikat) dafür, dass es keinen s-t-Weg geben kann, dessen Länge kürzer ist als $p(v_3) - p(v_1) = 1$. Ebenso beweist der Vektor $p = (0,4,2)^\mathsf{T}$, dass es keinen s-t-Weg geben kann, der Länge < 2 gibt.

Satz 2.7 gibt also eine Antwort auf die einleitende Frage: Um zu beweisen, dass ein s-t-Weg tatsächlich ein kürzester Weg ist, genügt es, ein Potential $p\colon V \to \mathbb{R}$ anzugeben, das $\ell(P) = p(t) - p(s)$ erfüllt. Die Prüfung, ob ein gegebener Vektor p ein Potential ist, ist effizient möglich. Sie erfordert die Prüfung von $|A|$ Ungleichungen, also i.A. viel weniger Arbeit, als das Aufzählen sämtlicher s-t-Wege.

BEMERKUNG 2.9. *Ein optimales Potential ist niemals eindeutig. Für jedes optimale Potential p, ist auch $p + C$ für $c \in \mathbb{R}$ ein optimales Potential.*

3. Berechnung kürzester Wege

Viele Algorithmen sind bekannt. Der vielleicht Bekannteste ist Dijkstras Algorithmus (VL Informatik I, Effiziente Algorithmen), für den Fall ausschließlich nichtnegativer Kantenlängen $\ell : A \to \mathbb{R}_{\geq 0}$.
Hier: Bellman-Ford Algorithmus, der beliebige Kantenlängen erlaubt, solange alle Kreise nichtnegative Länge besitzen – der Algorithmus erlaubt es dabei jedoch praktischerweise auch, die Existenz von Kreisen negativer Länge zu *erkennen*.

ALGORITHMUS 3.1. *Algorithmus von Bellman und Ford*

Eingabe : *Gerichteter Graph* $D = (V, A)$, $n = |V|$, $s \in V$, $\ell \colon A \to \mathbb{R}$
Ausgabe : *Funktionen* $d_0, \ldots, d_n \colon V \to (\mathbb{R} \cup \{\infty\})$, $g \colon V \setminus \{s\} \to V$
Setze $d_0(s) = 0$.
Setze $d_0(v) = \infty \ \forall v \in V \setminus \{s\}$.
for $k = 0$ ***to*** $n - 1$ **do**
  $d_{k+1}(v) = d_k(v) \ \forall v \in V$
  for $(u, v) \in A$ **do**
    if $d_{k+1}(v) > d_k(u) + \ell(u, v)$ **then**
      $d_{k+1}(v) = d_k(u) + \ell(u, v)$
      $g(v) = u$
    end
  end
end
if $d_n \neq d_{n-1}$ **then**
  Ausgabe*: „Es gibt einen Kreis negativer Länge, der von s aus erreichbar ist“.*
end

ANMERKUNG. *Streng genommen ist g auch für nicht von s aus erreichbare Knoten nicht definiert.*

BEMERKUNG 3.2. *Es lohnt sich an dieser Stelle ggf. ein Blick zurück auf Bemerkung 2.5. Man kann den Bellman-Ford-Algorithmus auch so darstellen, dass das Terminierungskriterium gerade ist, ein Potential erzeugt zu haben (siehe u.a. Vorlesung „Effiziente Algorithmen“).*

SATZ 3.3. *Es gilt*

$$d_k(v) = \min\{\ell(P) : P \text{ ist } s\text{-}v\text{-Kantenfolge, die höchstens } k \text{ Kanten enthält}\}.$$

SATZ 3.4. *Nach Ablauf des Algorithmus von Bellman und Ford gilt $d_n = d_{n-1}$ genau dann, wenn alle von s aus erreichbaren Kreise nicht-negative Länge haben.*

BEMERKUNG 3.5.

(a) *Die Laufzeit des Algorithmus von Bellman und Ford ist proportional zu $|V| \cdot |A| \leq |V|^3$.*
(b) *Falls alle von s aus erreichbaren Kreise nicht-negative Länge besitzen, dann ist* $\operatorname{dist}(s, v) = d_{n-1}(v)$ *für alle $v \in V$. Außerdem sind*

$$v, g(v), g(g(v)), \ldots, s$$

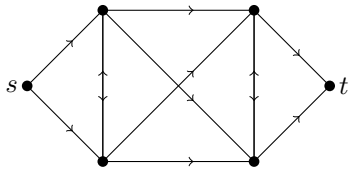
die Knoten eines kürzesten $s - v$-Weges in umgekehrter Reihenfolge.

Als abschließende Bemerkung dieses Kapitels sei erwähnt, dass sich die Grundidee der „tabellenartigen“ Distanz- (oder allgemeiner Wert-) Berechnung noch stark verallgemeinern und auf eine wesentlich breitere (und schwierigere) Klasse von Optimierungsproblemen anwenden lässt (dann jedoch häufig nicht notwendigerweise mehr mit einer polynomiellen Laufzeit). Das entsprechende Konzept heißt „Dynamische Programmierung“, und wird z.B. in der Vorlesung „Algorithmen der linearen und diskreten Optimierung“ kurz umrissen.

KAPITEL II

Flüsse in Netzwerken

In diesem Kapitel wenden wir uns Transportproblemen in einem Netzwerk zu. Das Ziel wird sein möglichst viele Güter zwischen zwei festgelegten Orten zu transportieren. Das könnten z.B. Flüssigkeit in einer Pipeline, Strom in einem Energienetzwerk, Autos in einem Strassennetz, oder Menschen in einem Flughafen sein.



Das Netzwerk sollte man sich vielleicht als ein System von Röhren vorstellen, wobei die zu transportierenden Güter in jeder Röhre nur in eine Richtung fließen dürfen. Im Röhrensystem sind zwei Knoten ausgezeichnet, die Quelle s und die Senke t. Nur an der Quelle können Güter in das Röhrensystem kommen und nur an der Senke können sie wieder entnommen werden. An jedem anderen Knoten der Röhrensystems müssen ankommende und weitergeleitete Güter im Gleichgewicht sein. Jedes Rohr habe eine gewisse Kapazität. Nun ist die Aufgabe einen möglichst großen Fluss von s nach t zu finden.
Dieses Optimierungsproblem wollen wir im Folgenden formalisieren.

1. Grundbegriffe

DEFINITION 1.1. *Es sei $D = (V, A)$ ein gerichteter Graph. Seien s, $t \in V$ zwei Knoten. Wir nennen s* Quelle *(engl. source) und t* Senke *(engl. terminal).*

(1) *Eine Funktion $f\colon A \to \mathbb{R}_{\geq 0}$ heißt s-t-*Fluss*, falls das* Flusserhaltungsgesetz

$$\sum_{a \in \delta^{in}(v)} f(a) = \sum_{a \in \delta^{out}(v)} f(a)$$

für alle Knoten $v \in V \setminus \{s, t\}$ erfüllt ist, wobei

$$\delta^{in}(v) = \{(w, v) \in A : w \in V\},$$
$$\delta^{out}(v)) = \{(v, w) \in A : w \in V\}.$$

(2) *Der* Wert *eines s-t-Flusses ist*

$$\mathit{value}(f) = \sum_{a \in \delta^{out}(s)} f(a) - \sum_{a \in \delta^{in}(s)} f(a) \quad \Big(= \sum_{a \in \delta^{in}(t)} f(a) - \sum_{a \in \delta^{out}(t)} f(a) \Big)$$

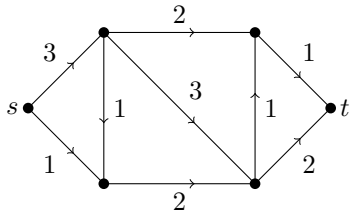
(3) *Ein s-t-Fluss f heißt* beschränkt *durch eine* Kapazitätsfunktion $c\colon A \to \mathbb{R}_{\geq 0}$*, falls $f(a) \leq c(a)$ für alle Kanten $a \in A$ gilt. Notation: $f \leq c$.*

(4) *Das Problem des Findens eines maximalen s-t-Flusses (*Maximum-Flow-Problem*) ist wie folgt definiert. Gegeben sei ein gerichteter Graph $D = (V, A)$, eine Quelle $s \in V$, eine Senke $t \in V$ und eine Kapazitätsfunktion $c\colon A \to \mathbb{R}_{\geq 0}$. Gesucht ist die Lösung des Maximierungsproblems:*

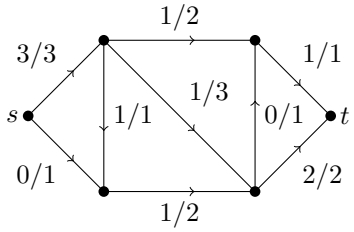
$$\max\{\ \mathit{value}(f) : f\colon A \to \mathbb{R}_{\geq 0} \text{ ist } s\text{-}t\text{-Fluss}, f \leq c\}$$

BEMERKUNG 1.2. *Es ist klar, dass das Maximum existiert, und einerseits (von unten) durch* 0 *sowie (von oben) durch* $\sum_{a\in\delta^{out}(s)} c(a)$ *beschränkt ist.*

BEISPIEL 1.3. *Ein Flussnetzwerk mit* $value(f) \in [0,3]$ *für alle* $f \leq c$.



Ein Fluss f *mit* $value(f) = 3$ *(Notation:* f_a/c_a*)*



Wie schon in Kapitel I wird eine Min-Max-Charakterisierung nützlich sein. Das zum Max-Flow-Problem zugehörige Minimierungsproblem ist das Min-Cut-Problem, welches wir nun definieren.

DEFINITION 1.4. *Es sei* $D = (V, A)$ *ein gerichteter Graph mit einer Quelle* $s \in V$ *und einer Senke* $t \in V$.

(1) *Die Teilmenge* $U \subseteq V$ *definiert die Schnitte*

$$\delta^{in}(U) = \{(v,u) \in A : u \in U,\ v \in V \setminus U\},\ \textit{und}$$
$$\delta^{out}(U) = \{(u,v) \in A : u \in U,\ v \in V \setminus U\}.$$

(2) *Falls* $s \in U$ *und* $t \in V \setminus U$ *ist, dann heißen* $\delta^{in}(U)$ *und* $\delta^{out}(U)$ *s-t-*Schnitte.

(3) *Es sei* $c\colon A \to \mathbb{R}_{\geq 0}$ *eine Kapazitätsfunktion. Dann ist*

$$c(\delta^{out}(U)) = \sum_{a\in\delta^{out}(U)} c(a)$$

die Kapazität des Schnittes $\delta^{out}(U)$*. Analog kann man* $c(\delta^{in}(U))$ *definieren. Es wird aber hier nicht benötigt.*

(4) *Das Problem des Findens eines minimalen s-t-Schnittes (*Min-Cut-Problem*) ist wie folgt definiert. Gegeben sei ein gerichteter Graph* $D = (V, A)$*, eine Quelle* $s \in V$*, eine Senke* $t \in V$ *und eine Kapazitätsfunktion* $c\colon A \to \mathbb{R}_{\geq 0}$*. Gesucht ist die Lösung des Minimierungsproblems*

$$\min\{c(\delta^{out}(U)) : U \subseteq V,\ s \in U,\ t \in V \setminus U\}.$$

2. Das Max-Flow-Min-Cut-Theorem

Wir können nun das Max-Flow-Min-Cut Theorem von Ford und Fulkerson formulieren, eine weitere Min-Max-Charakterisierung. Dieser Satz gehört zu den wichtigsten Aussagen des Operations Research.

SATZ 2.1. *(Max-Flow-Min-Cut-Theorem; Ford-Fulkerson, 1954)*
Es seien ein gerichteter Graph $D = (V, A)$, zwei Knoten $s, t \in V$ und eine Kapazitätsfunktion $c\colon A \to \mathbb{R}_{\geq 0}$ gegeben. Dann gilt

$$\begin{array}{lcl} \max \mathit{value}(f) & = & \min c(\delta^{out}(U)) \\ f \text{ ist } s\text{-}t\text{-Fluss} & & U \subseteq V \\ f \leq c & & s \in U,\ t \in V \setminus U \end{array}$$

Wichtiger Zusatz: Falls c ganzzahlig ist, das heißt $c(a) \in \mathbb{Z}$ für alle $a \in A$, dann gibt es einen ganzzahligen maximalen s-t-Fluss $f : A \to \mathbb{Z}_{\geq 0}$.

Der Zusatz ist sowohl von theoretischer als auch von praktischer Bedeutung. In der Praxis hat man es oft mit zu transportierenden Gütern zu tun, die man nicht teilen kann, so dass man ausschließlich mit ganzzahligen Flüssen arbeiten möchte.
Wir werden Satz 2.1 in diesem Kapitel schrittweise beweisen. Wir beginnen mit folgendem Lemma.

LEMMA 2.2. *Es seien ein gerichteter Graph $D = (V, A)$, zwei Knoten $s,\ t \in V$ und eine Kapazitätsfunktion $c\colon A \to \mathbb{R}_{\geq 0}$ gegeben. Sei f ein s-t-Fluss, der durch c beschränkt ist. Sei $U \subseteq V$ mit $s \in U, t \in V \setminus U$, so dass $\delta^{out}(U)$ ein s-t-Schnitt ist. Dann gilt*

$$(*) \qquad \mathit{value}(f) \leq c(\delta^{out}(U))$$

Es gilt Gleichheit in $(*)$ genau dann, wenn

$$f(a) = c(a) \quad \text{für alle } a \in \delta^{out}(U),$$
$$f(a) = 0 \quad \text{für alle } a \in \delta^{in}(U)$$

Ähnlich wie zuvor bei den kürzesten Wegen suchen wir nun nach einem Kriterium, mit dessen Hilfe wir feststellen können, ob ein gegebener s-t-Fluss noch verbessert werden kann.

DEFINITION 2.3. *Es seien ein gerichteter Graph $D = (V, A)$, $s,\ t \in V$, eine Kapazitätsfunktion $c\colon A \to \mathbb{R}_{\geq 0}$, und ein s-t-Fluss $f \leq c$ gegeben. Definiere den* Residualgraph *(bzw. das* Residualnetzwerk*) $D_f = (V, A_f)$ durch*

$$A_f = \{a \in A : f(a) < c(a)\} \cup \{a^{-1} \in A^{-1} : f(a) > 0\}.$$

wobei für eine Kante $a = (u, v)$

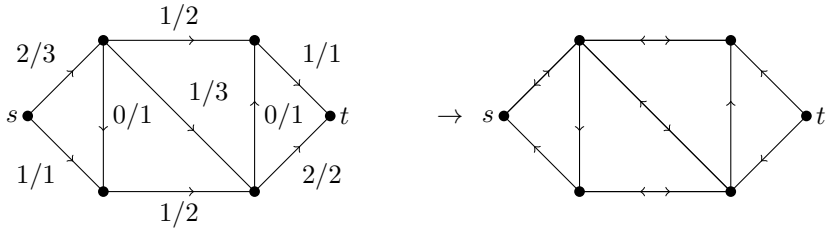
$$a^{-1} = (v, u) \text{ und } A^{-1} = \{a^{-1} : a \in A\}$$

sei.

BEMERKUNG 2.4. *Die Kanten eines Residualnetzwerks D_f können auch wieder mit den durch den Fluss f in D implizierten Kapazitäten versehen werden. Die Kapazitätsfunktion $c_f : A_f \to \mathbb{R}_{\geq 0}$ ist definiert durch*

$$c_f(a) = \begin{cases} c(a) - f(a) & , \text{ falls } a \in A \\ f(a) & , \text{ falls } a \in A^{-1}. \end{cases}$$

BEISPIEL 2.5. *Ein Fluss f in D (links) und das zugehörige Residualnetzwerk D_f (rechts).*



LEMMA 2.6. *Seien $D = (V, A)$, s, $t \in V$, $c\colon A \to \mathbb{R}_{\geq 0}$, und ein s-t-Fluss $f \leq c$ gegeben. Angenommen der Residualgraph D_f enthält keinen gerichteten s-t-Weg. Sei $U \subseteq V$ die Menge der Knoten, die in D_f von s aus erreichbar sind. Dann gilt*

$$\text{value}(f) = c(\delta^{out}(U)).$$

Insbesondere ist f nach Lemma 2.2 maximal.

Wir benötigen nun noch eine Definition, dann haben wir alle Werkzeuge zur Hand, um Satz 2.1 zu beweisen.

DEFINITION 2.7. *Seien $D = (V, A)$, s, $t \in V$, $c\colon A \to \mathbb{R}_{\geq 0}$, und ein s-t-Fluss $f \leq c$ sowie das zugehörige Residualnetzwek D_f gegeben. Angenommen es gibt einen s-t-Weg P in D_f. Definiere den* charakteristischen Vektor *$\chi^P\colon A \to \mathbb{R}$ durch*

$$\chi^P(a) = \begin{cases} 1, & \text{falls } P \text{ die Kante } a \text{ durchläuft,} \\ -1, & \text{falls } P \text{ die Kante } a^{-1} \text{ durchläuft,} \\ 0, & \text{sonst.} \end{cases}$$

BEMERKUNG 2.8. *Man beachte, dass der s-t-Weg P in Definition 2.7 nicht sowohl eine Kante a als auch a^{-1} zugleich enthalten kann, da P sonst kein (einfacher) Weg wäre.*

3. Berechnung maximaler s-t-Flüsse

ALGORITHMUS 3.1. *Algorithmus von Ford und Fulkerson (1955)*

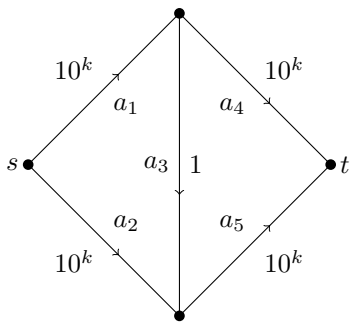
Eingabe : *Gerichteter Graph $D = (V, A)$, $s, t \in V$, $c\colon A \to \mathbb{R}_{\geq 0}$*
Ausgabe : *Maximaler s-t-Fluss f*
Setze $f = 0$
while *$\exists$ gerichteter s-t-Weg P in D_f* **do**
  $f = f + \varepsilon\chi^P$, wobei $\varepsilon > 0$ maximal gewählt, so dass $0 \leq f + \varepsilon\chi^P \leq c$ gilt
end

SATZ 3.2. *Falls $c(a) \in \mathbb{Q}$ für alle $a \in A$, dann terminiert der Ford-Fulkerson Algorithmus nach endlich vielen Schritten.*

BEMERKUNG 3.3. *Es gibt Beispiele mit $c(a) \in \mathbb{R}$, so dass der Algorithmus nicht terminiert. (→ siehe Schrijver - CO - Buch, 10.4a)*

BEMERKUNG 3.4. *Auch für rationale oder ganzzahlige Kapazitäten kann die Anzahl der Iterationen des Algorithmus von Ford und Fulkerson in seiner Ursprungsform jedoch unnötig hoch werden. Wie das folgende Beispiel zeigt, liegt dies daran, dass die Anzahl der Iterationen von der Wahl der Wege abhängt, entlang derer der Flusswert erhöht wird - und es für diese Wahl keine Vorschrift gibt.*

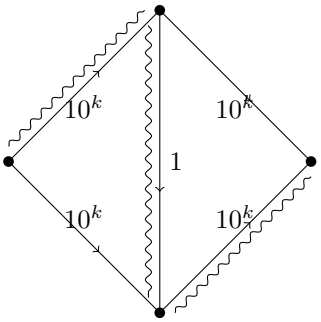
BEISPIEL 3.5.



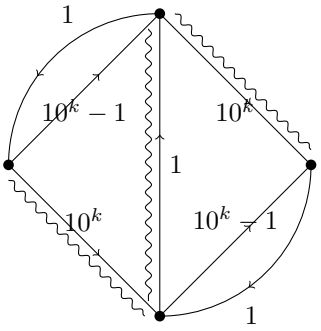
Offensichtlich ist $f = (f(a_1), f(a_2), f(a_3), f(a_4), f(a_5)) = (10^k, 10^k, 0, 10^k, 10^k)$ *der* s-t*-Fluss mit maximalem Wert für dieses Netzwerk mit value*$(f) = 2 \cdot 10^k$.

Aber: wählt man die Wege im Ford-Fulkerson Algorithmus sehr ungünstig, dann wird der Wert des Flusses in jeder Iteration nur um 1 *erhöht. Man könnte also insgesamt bis zu* $2 \cdot 10^k$ *Iterationen durchführen, wie wir nun sehen werden (obwohl man im günstigsten Fall mit zwei Iterationen auskäme). Der Fluss wird immer entlang der Schlangenlinien verbessert:*

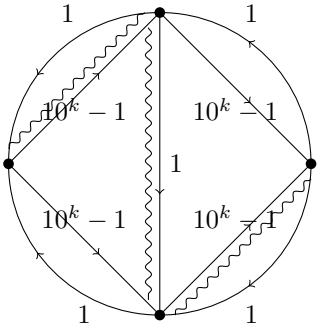
$f_1 = (0, 0, 0, 0, 0)$ $\quad D_{f_1}$*:*



$f_2 = (1, 0, 1, 0, 1)$ $\quad D_{f_2}$*:*



$f_3 = (1, 1, 0, 1, 1)$ $\quad D_{f_3}$*:*



$f_4 = (2, 1, 1, 1, 2)$

$\vdots$

Und so weiter.

Lösung dieses Problems (Dinits (1970), Edmonds-Karp (1972)): So etwas kann nicht passieren, wenn die gewählten s-t-Wege minimale Länge (bezogen auf die Anzahl der Kanten) haben.

LEMMA 3.6. *Es seien* $D = (V, A)$ *ein gerichteter Graph,* $s, t \in V$ *zwei Knoten und* $\mu(D) := \operatorname{dist}(s, t) < \infty$. *Sei* $\alpha(D) \subseteq A$ *die Menge der Kanten, die in einem kürzesten* s-t*-Weg vorkommen. Definiere* $D' = (V, A \cup \alpha(D)^{-1})$. *Dann gilt* $\mu(D') = \mu(D)$ *und* $\alpha(D') = \alpha(D)$.

Lemma 3.6 gilt für allgemeine gerichtete Graphen bei ausschließlichem Hinzufügen der Residualkanten zu Kanten auf kürzesten Wegen. Im Fall eines Residualnetzwerks fügen wir nach einer Flusserhöhung von f zu f' jedoch nicht nur Kanten hinzu, sondern entfernen auch mindestens eine (nämlich alle *saturierten* Kanten, d.h. jene, deren Kapazität von f' nun voll ausgenutzt ist). Dies kann jedoch erst recht nicht zu *kürzeren* s-t-Wegen oder neuen kürzesten s-t-Wegen führen. Die Länge kürzester Wege kann also nur entweder gleich bleiben, oder länger werden, aus $\mu(D') = \mu(D)$ wird $\mu(D_{f'}) \geq \mu(D_f)$.

KOROLLAR 3.7. *Seien $D = (V, A)$, s, $t \in V$, $c\colon A \to \mathbb{R}_{\geq 0}$, und ein s-t-Fluss $f \leq c$ gegeben. Sei $D_f = (V, A_f)$ der zugehörige Residualgraph, und angenommen, wir erhöhen f auf den Kanten eines kürzesten Weges in D_f. Sei f' der neue s-t-Fluss und $D_{f'} = (V, A_{f'})$ das zugehörige Residualnetzwerk. Dann gilt $\mu(D_{f'}) \geq \mu(D_f)$.*

Bei Gleichheit ($\mu(D_{f'}) = \mu(D_f)$) gilt zudem $\alpha(D_{f'}) \subsetneq \alpha(D_f)$, ansonsten (im Fall $\mu(D_{f'}) > \mu(D_f)$) können die neuen Kanten natürlich Teil der neuen (jetzt längeren) kürzesten Wege sein. Dies genügt um die Anzahl der Iterationen des Algorithmus von Ford und Fulkerson wie folgt zu beschränken.

SATZ 3.8. *Falls in jeder Iteration des Ford-Fulkerson Algorithmus ein kürzester s-t-Weg in D_f ausgewählt wird, beträgt die Anzahl der Iterationen höchstens $|V| \cdot |A|$. Insbesondere ist sie unabhängig von der Kapazitätsfunktion.*

KAPITEL III

Matchings in bipartiten Graphen

Motivation: Finden optimaler (injektiver, order ggf. sogar bijektiver) Abbildungen bzw. eineindeutiger Zuordnungen. Zum Beispiel:

- WG-Zimmern zu Studierenden
- Terminal-Gates zu Flugzeugen
- Schichten zu Mitarbeitern
- ...

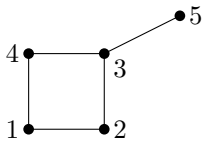
Dazu beschäftigen wir uns jetzt mit *ungerichteten* Graphen, also Graphen, deren Kanten keine Orientierung besitzen. Die meisten Begriffe definiert man sehr ähnlich wie im gerichteten Fall, dennoch beginnen wir zunächst mit etwas Notation.

1. Grundbegriffe

DEFINITION 1.1. *Ein* ungerichteter Graph *ist ein Paar $G = (V, E)$, bestehend aus einer endlichen Menge V, der Menge der* Knoten *von G, und aus einer Menge*

$$E \subseteq \{\{v, w\} : v, w \in V, v \neq w\},$$

der Menge der ungerichteten Kanten *von G. Mit anderen Worten, ein Graph G beschreibt eine symmetrische, irreflexive Relation auf $V \times V$.*



$V = \{1, \ldots, 5\}$
$E = \{\{1,2\}, \{2,3\}, \{3,4\}, \{4,1\}, \{3,5\}\}$

Man definiert nun die Begriffe *Kantenfolge*, *Weg* und *Kreis* genauso wie bei gerichteten Graphen. Ebenso schreiben wir manchmal wieder Funktionen $f\colon E(V) \to \mathbb{R}$ als $f \in \mathbb{R}^E(\mathbb{R}^V)$.

DEFINITION 1.2. *Es sei $G = (V, E)$ ein ungerichteter Graph, und v, $w \in V$.*

(1) *Die Knoten v und w heißen* benachbart (adjazent)*, falls $\{v, w\} \in E$. Dann heißt v* Nachbar *von w, und umgekehrt.*
(2) *Die Knoten v und w heißen* wegzusammenhängend*, falls es einen v-w-Weg gibt.*
(3) *Die Relation „wegzusammenhängend“ ist eine Äquivalenzrelation. Die zugehörigen Äquivalenzklassen heißen* Zusammenhangskomponenten.
(4) *Falls G nur eine Zusammenhangskomponente besitzt, so heißt G* zusammenhängend.

DEFINITION 1.3. *Ein ungerichteter Graph $G = (V, E)$ heißt* bipartit*, falls es Teilmengen $U, W \subseteq V$ gibt, so dass*

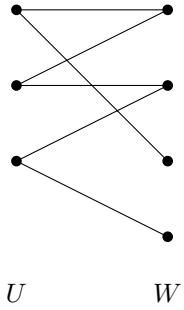
(1) *die Mengen U und W eine* (Bi-)Partition *von V sind, d.h.*

$$V = U \cup W, \quad U \cap W = \emptyset,$$

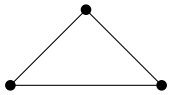
(2) *jede Kante von G genau einen Knoten aus U und einen aus W verbindet:*

$$|e \cap U| = |e \cap W| = 1 \quad \text{für alle } e \in E.$$

Beispiel 1.4.



Folgendes ist hingegen kein bipartiter Graph:



Tatsächlich kann man zeigen, dass ein Graph bipartit ist, genau dann, wenn er keinen Kreis ungerader Länge enthält.

2. Berechnung von Matchings mit maximaler Kardinalität

Definition 2.1. *Es sei $G = (V, E)$ ein ungerichteter Graph.*

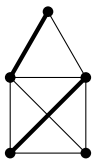
(1) *Ein* Matching *M in G ist eine Teilmenge disjunkter Kanten, das heißt*
$M \subseteq E$ *ist ein Matching in* $G \iff \forall e, f \in M, e \neq f : \ e \cap f = \emptyset$.

(2) *Die* Matchingzahl *von G ist*
$$\nu(G) = \max\{|M| : M \subseteq E \text{ ist Matching in } G\}.$$

(3) *Ein Matching heißt* perfekt*, falls $2|M| = |V|$ gilt.*

Im Fall einer Kante $e = \{v, w\} \in M$, M Matching, sagen wir auch, dass v und w *(M-)überdeckt* sind.

Beispiel 2.2. *In folgendem Graphen G gilt $\nu(G) = 2$, und er besitzt kein perfektes Matching.*

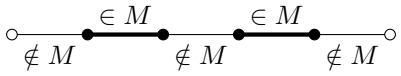


Wie schon im vorherigen Kapitel fassen wir einen Weg $P = (v_0, e_1, v_1, \ldots, e_m, v_m)$ eindeutig als eine Teilmenge der Kanten von G auf und schreiben vereinfachend $P = \{e_1, \ldots, e_m\} \subseteq E$.

Definition 2.3. *Es sei $G = (V, E)$ ein ungerichteter Graph und es sei $M \subseteq E$ ein Matching in G. Ein Weg $P \subseteq E$ heißt* M-augmentierend*, falls die folgenden zwei Bedingungen erfüllt sind:*

(1) *Weder Startknoten v_0 noch Endknoten v_m von P werden von M überdeckt: für alle $e \in M$ gilt $v_0 \notin e$ und $v_m \notin e$.*

(2) *Die Kanten $e_1, \ldots, e_m$ von P sind alternierend nicht aus M und aus M: $e_1 \notin M$, $e_2 \in M$, $e_3 \notin M$, $\ldots$, $e_{m-1} \in M$, $e_m \notin M$.*



LEMMA 2.4. *Es sei $G = (V, E)$ ein ungerichteter Graph und es sei $M \subseteq E$ ein Matching in G. Sei $P \subseteq E$ ein M-augmentierender Weg. Dann gilt:*

(1) *$|P|$ ist ungerade.*
(2) *Die symmetrische Differenz*

$$M' = M \Delta P = (M \setminus P) \cup (P \setminus M)$$

ist ein Matching in G mit $|M'| = |M| + 1$.

SATZ 2.5. *Es sei $G = (V, E)$ ein ungerichteter Graph und es sei M ein Matching in G. Dann hat M maximale Kardinalität genau dann, wenn es keinen M-augmentierenden Weg in G gibt.*

Satz 2.5 zeigt unmittelbar, dass der folgende Algorithmus korrekt ist.

ALGORITHMUS 2.6. *Bestimmung eines Matchings maximaler Kardinalität*

Eingabe : *Ungerichteter Graph $G = (V, E)$*
Ausgabe : *Matching $M \subseteq E$ mit $\nu(G) = |M|$*
$M = \emptyset$
while $\exists M$*-augmentierender Weg P* **do**
  $M = M \Delta P$.
end

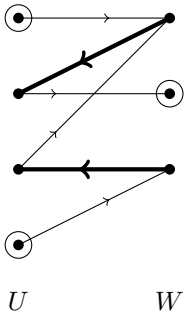
Der Algorithmus lässt jedoch offen, wie man einen M-augmentierenden Weg findet. Wenn man in den Beweis von Satz 2.5 schaut, dann erkennt man, dass dieser kein brauchbares Verfahren dafür liefert: Um einen M-augmentierenden Weg zu konstruieren, verwendet man die Existenz eines Matchings M', das mehr Kanten enthält als M. Ein solches Matching wollen wir aber gerade mit Hilfe eines M-augmentierenden Weges finden. Wie also aus diesem Henne-Ei-Problem entkommen?

- Hier: Fall G bipartit
- Allgemeiner Fall: „Blütenschrumpf"-Algorithmus von Jack Edmonds, siehe Vorlesung „Effiziente Algorithmen" oder Lehrbücher.

DEFINITION 2.7. *Sei $G = (V, E)$ ein bipartiter Graph mit Bipartition $V = U \cup W$ und $M \subseteq E$ ein Matching in G. Definiere den (gerichteten)* Residualgraph *$D_M = (V, A_M)$ durch*

$$A_M = \{(u, w) \in U \times W : \{u, w\} \in E \setminus M\} \cup \{(w, u) \in W \times U : \{u, w\} \in E \cap M\}.$$

Der Residualgraph D_M entsteht also aus dem Originalgraph G, indem man die Kanten E mit einer Richtung versieht. Wenn man den bipartiten Graph G so zeichnet, dass auf der linken Seite die Knotenmenge U und auf der rechten Seite die Knotenmenge W ist, dann wird eine Kante $\{u, w\} \in E$ zu einer „Vorwärtskante", wenn sie nicht zum Matching M gehört und sonst zu einer „Rückwärtskante".



SATZ 2.8. *Sei $G = (V, E)$ ein bipartiter Graph mit Bipartition $V = U \cup W$ und $M \subseteq E$ ein Matching in G. Seien $U_M \subseteq U$, $W_M \subseteq W$ die Knoten, die* nicht *von M überdeckt werden. Dann entspricht jeder gerichtete Weg von einem Knoten in U_M zu einem Knoten in W_M einem M-augmentierenden Weg und umgekehrt.*

Algorithmische Umsetzung: Finde kürzeste Wege zwischen je zwei Knoten in U_M und W_M (z.B. mit mit Bellman-Ford), wobei $\ell\colon A_M \to \mathbb{Z}$, $\ell(a) = 1$ gesetzt wird. Dies führt zu einer polynomiellen, jedoch nicht bestmöglichen Laufzeit. Besser: Führe Augmentierungen auf einer maximalen Menge knoten-disjunkter Pfade gleicher Länge auf effiziente Weise „auf einmal“ durch (Ansatz von Hopcroft und Karp).

3. Das Matchingtheorem von König

Ziel dieses Abschnittes ist es, eine Min-Max-Charakterisierung für die Matchingzahl $\nu(G)$ eines bipartiten Graphen G zu finden.

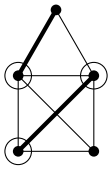
DEFINITION 3.1. *Sei $G = (V, E)$ ein ungerichteter Graph.*

(1) *Eine Teilmenge $C \subseteq V$ heißt* Knotenüberdeckung *von G, falls jede Kante einen Knoten aus C enthält, d.h.*

$$\forall e \in E : |C \cap e| \geq 1.$$

(2) *Die* Knotenüberdeckungszahl *von G ist definiert als*

$$\tau(G) = \min\{|C| : C \subseteq V \text{ ist Knotenüberdeckung von } G\}.$$



SATZ 3.2. *(Das Matchingtheorem von König, 1931)*
Sei $G = (V, E)$ ein bipartiter Graph. Dann ist die Matchingzahl von G gleich der Knotenüberdeckungszahl von G, d.h.

$$\nu(G) = \tau(G).$$

BEMERKUNG 3.3. *Im Beweis sieht man, dass die Ungleichung $\nu(G) \leq \tau(G)$ nicht nur für bipartite Graphen sondern auch für allgemeine Graphen gilt. Dagegen gilt die andere Ungleichung $\nu(G) \geq \tau(G)$ im allgemeinen Fall nicht, wie man sich am Beispiel eines Dreiecks verdeutlicht. Desweiteren gibt der Beweis ein konstruktives und effizientes Verfahren an, um eine optimale Knotenüberdeckung für bipartite Graphen G zu* bestimmen.

KOROLLAR 3.4. *(P. Hall, 1935)*
Sei $G = (V, E)$ ein bipartiter Graph mit Bipartition $V = U \cup W$. Dann gilt $\nu(G) = |U|$ genau dann, wenn für alle Teilmengen $X \subseteq U$ gilt

$$|\Gamma(X)| \coloneqq |\{w \in W : \{x, w\} \in E, x \in X\}| \geq |X|.$$

Aus dem Satz von Hall folgt unmittelbar der sogenannte Heiratssatz, der Aufschluss darüber gibt, unter welchen Voraussetzungen ein bipartiter Graph ein perfektes Matching besitzt.

KOROLLAR 3.5. *('Marriage Theorem', nach H. Weyl, 1949)*
Sei $G = (V, E)$ ein bipartiter Graph. Sei $V = U \cup W$ eine Bipartition. Dann gibt es ein perfektes Matching in G genau dann, wenn $|U| = |W|$ ist und für jede Teilmenge $X \subseteq U$ gilt $|\Gamma(X)| \geq |X|$.

4. Matchings mit maximalem Gewicht: Die ungarische Methode

Im Folgenden beschäftigen wir uns mit Graphen, bei denen die Kanten Gewichte besitzen. Ziel wird es sein, ein maximales gewichtetes Matching zu bestimmen.

DEFINITION 4.1. *Sei $G = (V, E)$ ein ungerichteter Graph und $w\colon E \to \mathbb{R}$ eine* Gewichtsfunktion.

(1) *Für $M \subseteq E$ definiere*

$$w(M) = \sum_{e \in M} w(e)$$

als das Gewicht von M.

(2) *Die* gewichtete Matchingzahl *von G und w ist*

$$\nu_w(G) = \max\{w(M) : M \subseteq E \text{ Matching in } G\}.$$

DEFINITION 4.2. *Sei $G = (V, E)$ ein ungerichteter Graph, $w\colon E \to \mathbb{R}$ eine Gewichtsfunktion, und $M \subseteq E$ ein Matching. M heißt* extrem *(für G und w), falls für alle Matchings $M' \subseteq E$ mit $|M'| = |M|$ gilt, dass $w(M') \leq w(M)$.*

DEFINITION 4.3. *Sei $G = (V, E)$ ein ungerichteter Graph, $w\colon E \to \mathbb{R}$ eine Gewichtsfunktion, und $M \subseteq E$ ein Matching. Definiere die Längenfunktion $\ell_M\colon E \to \mathbb{R}$ durch*

$$\ell_M(e) = \begin{cases} w(e), & \text{falls } e \in M \\ -w(e), & \text{falls } e \notin M. \end{cases}$$

Für $P \subseteq E$ definiere

$$\ell_M(P) = \sum_{e \in P} \ell_M(e).$$

SATZ 4.4. *Sei $G = (V, E)$ ein ungerichteter Graph, $w\colon E \to \mathbb{R}$ eine Gewichtsfunktion, und $M \subseteq E$ ein extremes Matching für G und w. Sei $P \subseteq E$ ein M-augmentierender Weg minimaler Länge bzgl. ℓ_M. Dann ist auch $M' = M \Delta P$ ein extremes Matching für G und w.*

ALGORITHMUS 4.5. *„Ungarische Methode" von Egerváry (1931)*

Eingabe : *Ungerichteter Graph $G = (V, E)$, $w\colon E \to \mathbb{R}$ Gewichtsfunktion*
Ausgabe : $\nu_w(G)$
$M_0 = \emptyset$
$k = 1$
while $\exists M_{k-1}$*-augmentierender Weg* **do**
  Wähle M_{k-1}-augmentierenden Weg P mit minimaler Länge bzgl. $\ell_{M_{k-1}}$.
  $M_k = M_{k-1} \Delta P$
  $k = k + 1$
end
output $\max\{w(M_i) : i = 0, 1, \ldots, k-1\}$.

SATZ 4.6. *Algorithmus 4.5 ist korrekt, d.h. berechnet tatsächlich $\nu_w(G)$.*

Auch hier bleibt wieder zu klären, wie wir für einen bipartiten Graphen $G = (V, E)$ mit Partitionen $V = U \cup W$, und ein Matching $M \subseteq E$, jeweils einen M-augmentierenden Weg mit minimaler Länge bzgl. ℓ_M finden.
Dazu betrachten wir wieder den Residualgraph $D_M = (V, A_M)$ (nun mit $M = M_{k-1}$) und versehen ihn, folgend Definition 4.3, mit der auf A_M erweiterten Kantenlängenfunktion $\ell'_M\colon A_M \to \mathbb{R}$, definiert durch

$$\ell'_M((a, b)) = \ell_M(\{a, b\}).$$

Nun entspricht die Bestimmung eines M_{k-1}-augmentierenden Weges mit minimaler Länge bzgl. $\ell_{M_{k-1}}$ wieder dem Finden einen kürzesten Weges von U_M nach W_M. Dieser kann wie bisher bestimmt werden, da D_M keine gerichteten Kreise negativer Länge besitzt, wie der folgende Satz zeigt:

SATZ 4.7. *Sei nun $G = (V, E)$ ein bipartiter Graph und $w\colon E \to \mathbb{R}$ eine Gewichtsfunktion. Sei $M \subseteq E$ ein extremes Matching für G und w. Dann besitzt der Residualgraph D_M mit Kantenlängenfunktion ℓ_M keine gerichteten Kreise negativer Länge bzgl. ℓ_M.*

Teil B

Einführung in die konvexe und lineare Optimierung

KAPITEL IV

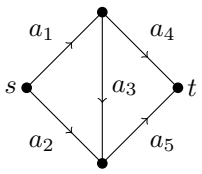
Konvexität

Viele Optimierungsprobleme des Operations Research lassen sich als konvexe bzw. als lineare Programme formulieren.
Gegeben: $c \in \mathbb{R}^n$, $a_1, \ldots, a_m \in \mathbb{R}^n$, $b_1, \ldots, b_m \in \mathbb{R}$
Gesucht: $\max\{c^\mathsf{T} x : x \in \mathbb{R}^n, a_j^\mathsf{T} x \leq b_j \text{ für alle } j = 1, \ldots, m\}$

Der Vektor c definiert die (lineare) Zielfunktion $x \mapsto c^\mathsf{T} x$. Die linearen Ungleichungen $a_j^\mathsf{T} x \leq b_j$ sind die Nebenbedingungen.
Notation: $Ax \leq b$ mit $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, und

$$A = \begin{bmatrix} — \; a_1^\mathsf{T} \; — \\ \vdots \\ — \; a_m^\mathsf{T} \; — \end{bmatrix}, b = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}$$

Beispiel 0.1. *Maximum-Flow Problem:*



$$\begin{aligned} &\max f_1 + f_2 \\ &f_1, \ldots f_5 \in \mathbb{R}_{\geq 0} \\ &f_1, f_2, f_4, f_5 \leq 100 \\ &f_3 \leq 1 \\ &f_3 + f_4 = f_1 \\ &f_2 + f_3 = f_5 \end{aligned}$$

oder:

$$\begin{aligned} &\max (1,1,0,0,0)^\mathsf{T} f \\ &f \in \mathbb{R}^5_{\geq 0} \\ &\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ 1 & & -1 & -1 & & \\ -1 & & 1 & 1 & & \\ & 1 & 1 & & -1 & \\ & -1 & -1 & & 1 & \end{bmatrix} f \leq \begin{bmatrix} 100 \\ 100 \\ 1 \\ 100 \\ 100 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{aligned}$$

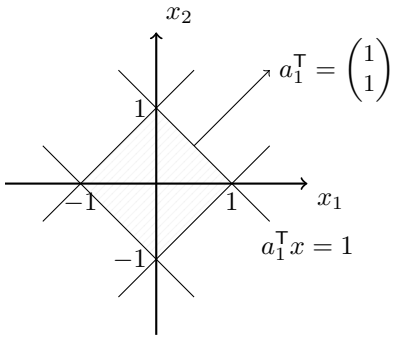
Auch die Bedingungen $f_i \geq 0$ lassen sich hier noch in die Matrixform einbauen.
Ziel:

- Finde geometrische Beschreibung der Menge $\{x \in \mathbb{R}^n : Ax \leq b\}$ der zulässigen Lösungen.
- Finde geometrische Erklärung für / von allgemeine/n Min-Max-Charakterisierungen.
- Verwende geometrische Beobachtungen, um Algorithmen für lineare Programme zu entwerfen.

Beispiel 0.2.

$$A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Geometrische Darstellung von $\{x \in \mathbb{R}^2 : Ax \leq b\}$*:*



Dies ist ein (konvexes) Polyeder.

Häufig in der Modellierung praxisnaher Optimierungsprobleme: Nicht-Negativitätsbedingungen. → Erweiterung der linearen Algebra über $\mathbb{R}$ durch diese.

1. Grundbegriffe aus Topologie und Geometrie

Generalvoraussetzung: Sei E ein n-dimensionaler euklidischer $\mathbb{R}$-Vektorraum mit Skalarprodukt $\langle x, y\rangle$ und Norm $\|x\| = \sqrt{\langle x, x\rangle}$. Aus der linearen Algebra ist bekannt, dass wir durch Auswahl einer Orthonormalbasis von E annehmen können, dass $E = \mathbb{R}^n$ und $\langle x, y\rangle = x^\mathsf{T} y$ ist.

Definition 1.1. (a) *Die* Kugel $B(x, r)$ mit Mittelpunkt $x \in \mathbb{R}^n$ und Radius $r \geq 0$ *ist definiert als*

$$B(x, r) = \{y \in \mathbb{R}^n : \|x - y\| \leq r\}.$$

(b) *Sei $A \subseteq \mathbb{R}^n$ eine Menge. Ein Punkt $x \in A$ heißt* innerer Punkt *von A, falls es ein $\varepsilon > 0$ gibt mit $B(x, \varepsilon) \subseteq A$. Das Innere von A ist*

$$\operatorname{int} A = \{x \in A : x \text{ innerer Punkt von } A\}.$$

(c) *Die Menge A heißt* offen*, falls $A = \operatorname{int} A$ ist.*
(d) *Die Menge A heißt* abgeschlossen*, falls $\mathbb{R}^n \setminus A$ offen ist.*
(e) *Der* Abschluss *von A ist*

$$\overline{A} = \bigcap_{B \supseteq A, B \text{ abgeschlossen}} B.$$

(f) *Die Menge A heißt kompakt, wenn jede Folge $(x_i)_{i\in\mathbb{N}}$ bestehend aus Elementen aus A eine konvergente Teilfolge mit Grenzwert in A besitzt.*
(g) *Die Menge A heißt beschränkt, falls ein $r \geq 0$ existiert, so dass $A \subseteq B(0, r)$.*

Bemerkung 1.2. *Es gilt:*

(i) *Die Menge A ist abgeschlossen $\Leftrightarrow$ jede konvergente Folge $(x_i)_{i\in\mathbb{N}}$ mit $x_i \in A$ besitzt Grenzwert $\lim_{i\to\infty} x_i \in A$.*
(ii) *A ist kompakt $\Leftrightarrow$ A ist abgeschlossen und beschränkt.*

Definition 1.3. *Sei $A \subseteq \mathbb{R}^n$. Der Rand von A ist definiert als*

$$\partial A = \{x \in \mathbb{R}^n : \forall \varepsilon > 0 : B(x, \varepsilon) \cap A \neq \emptyset \text{ und } B(x, \varepsilon) \cap (\mathbb{R}^n \setminus A) \neq \emptyset\}.$$

Bemerkung 1.4. *Es gilt:*

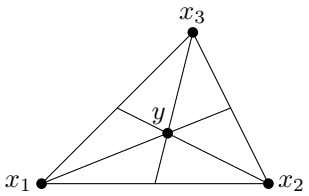
(i) *Der Rand ∂A von A ist abgeschlossen.*
(ii) *$\overline{A} = A \cup \partial A$ und $\partial A = \overline{A} \setminus (\operatorname{int} A)$.*

2. Konvexe Mengen und konvexe Funktionen

DEFINITION 2.1. *Seien $x_1, \dots, x_N \in \mathbb{R}^n$. Dann ist $y \in \mathbb{R}^n$ eine* Konvexkombination *von $x_1, \dots, x_N$, falls*

$$y = \sum_{i=1}^{N} \alpha_i x_i \quad \textit{mit } \alpha_1, \dots, \alpha_N \geq 0 \quad \textit{und } \sum_{i=1}^{N} \alpha_i = 1.$$

BEISPIEL 2.2. *Der Schwerpunkt eines Dreiecks ist eine Konvexkombination der drei Eckpunkte.*



$y = \frac{1}{3}x_1 + \frac{1}{3}x_2 + \frac{1}{3}x_3$

DEFINITION 2.3. *Eine Menge $C \subseteq \mathbb{R}^n$ heißt* konvex*, wenn jede Konvexkombination von Punkten aus C wieder einen Punkt aus C ergibt (die Menge C also unter der Bildung von Konvexkombinationen abgeschlossen ist), das heißt*

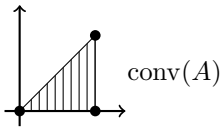
$$\forall N \in \mathbb{N}\, \forall x_1, \dots, x_N \in C\, \forall \alpha_1, \dots, \alpha_N \geq 0 : \sum_{i=1}^{N} \alpha_i = 1 \quad \Rightarrow \sum_{i=1}^{N} \alpha_i x_i \in C.$$

DEFINITION 2.4. *(konvexe Hülle) Sei $A \subseteq \mathbb{R}^n$. Die* konvexe Hülle *von A ist*

$$\operatorname{conv}(A) = \bigcap_{\substack{B \supseteq A \\ B \text{ konvex}}} B.$$

Da der Durchschnitt von konvexen Mengen wieder konvex ist, handelt es sich bei $\operatorname{conv}(A)$ um die inklusionsminimale konvexe Menge, die A enthält.

BEISPIEL 2.5. $A = \left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$



$\operatorname{conv}(A)$

Eine alternative Charakterisierung der konvexen Hülle bietet nachfolgender Satz.

SATZ 2.6. *Sei $A \subseteq \mathbb{R}^n$. Dann gilt*

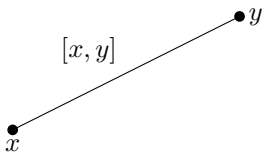
$$\operatorname{conv}(A) = \{y \in \mathbb{R}^n : \exists N \in \mathbb{N}, x_1, \dots, x_N \in A, \alpha_1, \dots, \alpha_N \geq 0, \sum_{i=1}^{N} \alpha_i = 1 : y = \sum_{i=1}^{N} \alpha_i x_i\}.$$

Das heißt, $\operatorname{conv}(A)$ besteht aus allen Konvexkombinationen der Elemente von A.

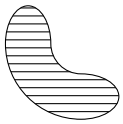
DEFINITION 2.7. *Seien $x, y \in \mathbb{R}^n$ Punkte, dann ist*

$$[x, y] = \operatorname{conv}\{x, y\} = \{(1 - \alpha)x + \alpha y : \alpha \in [0, 1]\}$$

die Verbindungsstrecke *zwischen x und y.*



SATZ 2.8. *Eine Menge $C \subseteq \mathbb{R}^n$ ist konvex $\Leftrightarrow\ \forall x, y \in C : [x, y] \subseteq C$.*

BEISPIEL 2.9. (a)  *ist keine konvexe Menge.*

(b) *Sei* $\|\cdot\|\colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ *eine Norm, und*

$$K = \{x \in \mathbb{R}^n : \|x\| \leq 1\}$$

die zugehörige Einheitskugel.

Behauptung: K ist konvex.

Beweis: Seien $x, y \in K$, $\alpha \in [0,1]$. *Dann ist*

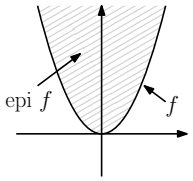
$$\|(1-\alpha)x+\alpha y\| \leq \|(1-\alpha)x\|+\|\alpha y\| = (1-\alpha)\underbrace{\|x\|}_{\leq 1} + \alpha\underbrace{\|y\|}_{\leq 1} \leq (1-\alpha)\cdot 1+\alpha\cdot 1 = 1.$$

Jede Konvexkombination ist also wieder in K.

DEFINITION 2.10. *Sei* $C \subseteq \mathbb{R}^n$ *eine konvexe Menge. Eine* Funktion $f\colon C \to \mathbb{R}$ *heißt* konvex*, wenn ihr Epigraph, der durch*

$$\operatorname{epi} f \coloneqq \{(x,\beta) \in C \times \mathbb{R} : f(x) \leq \beta\} \subseteq \mathbb{R}^{n+1}$$

definiert ist, eine konvexe Menge ist.

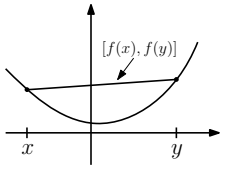


BEMERKUNG 2.11. *Die Funktion f aus Definition 2.10 heißt* konkav*, wenn* $-f$ *konvex ist.*

SATZ 2.12. *(Ungleichung von Jensen)*
Sei $C \subseteq \mathbb{R}^n$ *konvex. Eine Funktion* $f\colon C \to \mathbb{R}$ *ist genau dann konvex, wenn für alle* x, $y \in C$ *und alle* $\alpha \in [0,1]$ *stets die Ungleichung*

$$f((1-\alpha)x + \alpha y) \leq (1-\alpha)f(x) + \alpha f(y)$$

gilt.



DEFINITION 2.13. *Es seien* $C \subseteq \mathbb{R}^n$ *eine konvexe Menge und* $f\colon C \to \mathbb{R}$ *eine konvexe Funktion. Diese definieren ein* konvexes Optimierungsproblem

$$\inf\{f(x) : x \in C\}.$$

Die Menge C heißt die *Menge der zulässigen Lösungen*, die Funktion f heißt *Zielfunktion* des konvexen Optimierungsproblems.
Viele Optimierungsprobleme können als konvexe Optimierungsprobleme formuliert werden (siehe z.B. Vorlesung „Konvexe Optimierung“).

SATZ 2.14. *Seien $C \subseteq \mathbb{R}^n$ eine konvexe Menge und $f\colon C \to \mathbb{R}$ eine konvexe Funktion. Angenommen, $x_0 \in C$ ist ein lokales Minimum des konvexen Optimierungsproblems* $\inf\{f(x) : x \in C\}$*, d.h. es gibt ein $\varepsilon > 0$, so dass*

$$f(x_0) = \inf\{f(x) : x \in C \cap B(x_0, \varepsilon)\}$$

gilt. Dann ist x_0 auch ein globales Optimum, d.h. es gilt

$$f(x_0) = \inf\{f(x) : x \in C\}.$$

Satz 2.14 hat weitreichende Konsequenzen insbesondere für algorithmische Ansätze zur Lösung konvexer Optimierungsprobleme:

(1) Sog. *Abstiegsverfahren* finden globale Minima über eine Folge von Punkten $x_1, x_2, x_3, \cdots \in C$ mit $f(x_1) \geq f(x_2) \geq f(x_3) \geq \ldots$.
(2) Garantieren der Eigenschaft $x_i \in C$ kann algorithmisch jedoch schwierig sein.

DEFINITION 2.15. (1) *Der Punkt $y \in \mathbb{R}^n$ heißt* affine Kombination *der Punkte $x_1, \ldots, x_N \in \mathbb{R}^n$, falls es $\alpha_1, \ldots, \alpha_N \in \mathbb{R}$ gibt, so dass*

$$1 = \sum_{i=1}^{N} \alpha_i \quad \textit{und} \quad y = \sum_{i=1}^{N} \alpha_i x_i.$$

(2) *Die Punkte $x_1, \ldots x_N \in \mathbb{R}^n$ sind* affin unabhängig, *falls*

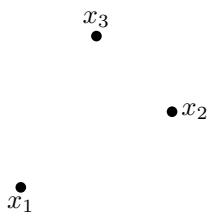
$$\forall \alpha_1, \ldots, \alpha_N \in \mathbb{R} : \sum_{i=1}^{N} \alpha_i = 0,\ \sum_{i=1}^{N} \alpha_i x_i = 0 \ \Rightarrow \alpha_1 = \ldots = \alpha_N = 0.$$

Alternativ:

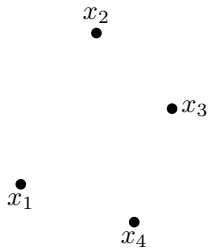
$$\forall \alpha_1, \ldots, \alpha_N \in \mathbb{R} : \sum_{i=1}^{N} \alpha_i = 0, \Rightarrow \sum_{i=1}^{N} \alpha_i x_i \neq 0 \textit{ oder } \alpha_1 = \ldots = \alpha_N = 0.$$

Oder: Die Vektoren $\begin{bmatrix} 1 \\ x_1 \end{bmatrix}, \ldots, \begin{bmatrix} 1 \\ x_N \end{bmatrix} \in \mathbb{R}^{n+1}$ *sind linear unabhängig.*

BEISPIEL 2.16. *Drei Punkte in der Ebene, die nicht alle auf der selben Gerade liegen, sind affin unabhängig.*



Vier Punkte in der Ebene sind hingegen nie affin unabhängig.



DEFINITION 2.17. *Eine Menge $A \subseteq \mathbb{R}^n$ heißt* affiner Unterraum*, falls jede affine Kombination von Punkten aus A wieder einen Punkt in A ergibt (die Menge A also unter der Bildung von affinen Kombinationen abgeschlossen ist), das heißt*

$$\forall N \in \mathbb{N}\, \forall x_1, \ldots, x_N \in A\, \forall \alpha_1, \ldots, \alpha_N \in \mathbb{R} : \sum_{i=1}^{N} \alpha_i = 1 \quad \Rightarrow \sum_{i=1}^{N} \alpha_i x_i \in A.$$

BEMERKUNG 2.18. *Affine Unterräume sind konvex.*

Affine Unterräume der Dimension 0 *sind Punkte.*
Affine Unterräume der Dimension 1 *sind Geraden.*
Allgemein: Affine Unterräume sind Hyperebenen. Mehr dazu im nächsten Abschnitt.

DEFINITION 2.19. *Die affine Hülle einer Menge $A \subseteq \mathbb{R}^n$ ist definiert als*

$$\operatorname{aff}(A) = \bigcap_{B \supseteq A, B \text{ affiner Unterraum}} B.$$

SATZ 2.20. *Für die affine Hülle einer Menge $A \subseteq \mathbb{R}^n$ gilt:*

$$\operatorname{aff}(A) = \left\{ y \in \mathbb{R}^n : \exists N \in \mathbb{N}, x_1, \ldots, x_N \in A, \alpha_1, \ldots, \alpha_N \in \mathbb{R}, \sum_{i=1}^{N} \alpha_i = 1 : y = \sum_{i=1}^{N} \alpha_i x_i \right\}.$$

DEFINITION 2.21. *Sei $A \subseteq \mathbb{R}^n$. Die* Dimension *von A ist*

$$\dim A = \dim \operatorname{aff}(A) = \max\{N - 1 : x_1, \ldots, x_N \in A \text{ affin unabhängig}\}.$$

BEMERKUNG 2.22. *Für $A \subseteq \mathbb{R}^n$ ist stets $\dim A \leq n$, weil* $\operatorname{rang} \begin{bmatrix} 1 & \ldots & 1 \\ x_1 & \ldots & x_N \end{bmatrix} \leq n + 1$. *Falls $A = \emptyset$ setzen wir $\dim A = -1$.*

DEFINITION 2.23. *Seien $x_1, \ldots, x_N \in \mathbb{R}^n$. Dann ist $y \in \mathbb{R}^n$ eine* konische Kombination *von $x_1, \ldots, x_N$, falls es $\alpha_1, \ldots, \alpha_N \geq 0$ gibt, so dass*

$$y = \sum_{i=1}^{N} \alpha_i x_i.$$

DEFINITION 2.24. *Eine Menge $C \subseteq \mathbb{R}^n$ heißt* konvexer Kegel*, falls für alle $\alpha, \beta \in \mathbb{R}$, $\alpha, \beta \geq 0$ und für alle $x, y \in C$ gilt:*

$$\alpha x + \beta y \in C.$$

D.h. C ist unter der Bildung von konischen Kombinationen abgeschlossen.

DEFINITION 2.25. *(konische Hülle)*
Sei $A \subseteq \mathbb{R}^n$. Die konische Hülle *von A ist*

$$\operatorname{cone}(A) = \bigcap_{\substack{B \supseteq A \\ B \text{ konvexer Kegel}}} B.$$

SATZ 2.26. *Sei $A \subseteq \mathbb{R}^n$, dann gilt*

$$\operatorname{cone}(A) = \{y \in \mathbb{R}^n : \exists N \in \mathbb{N}, \alpha_1, \ldots, \alpha_N \geq 0, x_1, \ldots, x_N \in A : y = \sum_{i=1}^{N} \alpha_i x_i\}$$

DEFINITION 2.27. *Ein konvexer Kegel $C \subseteq \mathbb{R}^n$ heißt* endlich erzeugt*, falls es $x_1, \ldots, x_N \in \mathbb{R}^n$ gibt mit*

$$C = \operatorname{cone}\{x_1, \ldots, x_N\}.$$

Wir sagen auch umgekehrt: $C = \operatorname{cone}\{x_1, \ldots, x_N\}$ ist der durch $x_1, \ldots, x_N$ erzeugte Kegel $C \subseteq \mathbb{R}^n$.

Beispiel 2.28. *Der nichtnegative Orthant*

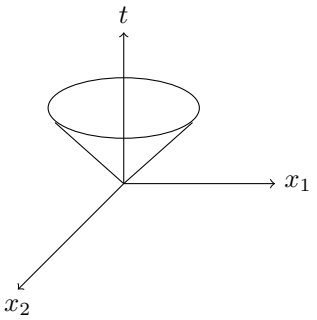
$$\mathbb{R}^n_{\geq 0} = \{x \in \mathbb{R}^n : x_1 \geq 0, \ldots, x_n \geq 0\} = \operatorname{cone}\{e_1, \ldots, e_n\}$$

ist ein endlich erzeugter Kegel.

Beispiel 2.29. *Der „Lorentzkegel“ oder „ice cream cone“*

$$\mathcal{L}^{n+1} = \{(x, t) \in \mathbb{R}^{n+1} : \|x\| \leq t\}$$

ist ein konvexer Kegel – aber nicht endlich erzeugt.



3. Trennungssätze

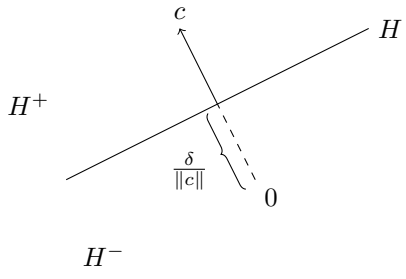
Sei $C \subseteq \mathbb{R}^n$, $C \neq \emptyset$, eine abgeschlossene, konvexe Menge.

Fundamentale Eigenschaft:
Jeder Punkt $z \notin C$ kann durch eine (affine) Hyperebene von C getrennt werden.

Die Trennung funktioniert nicht nur im $\mathbb{R}^n$, sondern auch in allg. lokal-konvexen topologischen Vektorräumen. Siehe Funktionalanalysis: Satz von Hahn-Banach
Wir leiten im Folgenden einen konstruktiven Beweis dieser Eigenschaft her.

Definition 3.1. *Eine Menge $H \subseteq \mathbb{R}^n$ heißt (*affine*)* Hyperebene*, falls es einen Vektor $c \in \mathbb{R}^n \setminus \{0\}$ und ein $\delta \in \mathbb{R}$ gibt mit*

$$H = \{x \in \mathbb{R}^n : c^\mathsf{T} x = \delta\}.$$

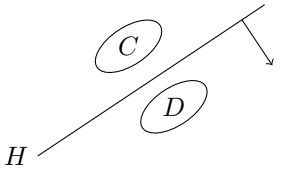


Definition 3.2. *Die abgeschlossenen und konvexen zugehörigen Mengen*

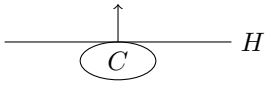
$$H^+ = \{x \in \mathbb{R}^n : c^\mathsf{T} x \geq \delta\}$$
$$H^- = \{x \in \mathbb{R}^n : c^\mathsf{T} x \leq \delta\}$$

heißen Halbräume.

DEFINITION 3.3. *Seien $C, D \subseteq \mathbb{R}^n$. Eine Hyperebene H heißt* Trennhyperebene *von C und D, falls $C \subseteq H^-$ und $D \subseteq H^+$ (oder umgekehrt).*

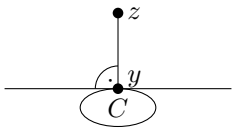


DEFINITION 3.4.
Sei $C \subseteq \mathbb{R}^n$. Eine Hyperebene H heißt Stützhyperebene *von C, falls $C \subseteq H^-$ und $C \cap H \neq \emptyset$.*



Frage: Wie findet man Trenn- bzw. Stützhyperebenen?

Antwort: Verwende *metrische Projektion*, d.h. die beste Approximation von $z \notin C$ in C. Sei $y \in C$ eine metrische Projektion: Nutze dann aus, dass Stützhyperebene von C an y gerade orthogonal zu $(z - y)$ liegt. (Metrische Projektion bei affinen Hyperebenen entspricht einer orthogonalen Projektion.)

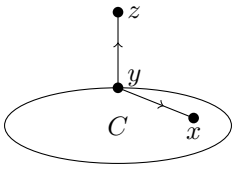


LEMMA 3.5. *Sei $C \subseteq \mathbb{R}^n$ abgeschlossen und konvex, $C \neq \emptyset$, und $z \notin C$. Dann gibt es genau einen Punkt $y \in C$ (bzw. genauer $y \in \partial C$) mit der Eigenschaft*

$$\|y - z\| = \inf\{\|x - z\| : x \in C\}.$$

Dieser Punkt wird auch als metrische Projektion von z auf C bezeichnet; Notation: $y = \pi_C(z)$. Darüber hinaus gilt für alle $x \in C$ die Ungleichung

$$(z - y)^\mathsf{T}(x - y) \leq 0.$$



BEMERKUNG 3.6. *Manchmal ist es hilfreich, die Definition von π_C auch auf Punkte $z \in C$ zu erweitern: Wir setzen natürlicherweise $\pi_C(z) = z$.*

SATZ 3.7. *Sei $C \subseteq \mathbb{R}^n$ abgeschlossen, konvex, und $C \neq \emptyset$. Sei $z \notin C$ ein Punkt außerhalb von C. Dann gibt es eine Trennhyperebene von $\{z\}$ und C.*

BEMERKUNG 3.8. *Die im Beweis konstruierte Hyperebene*

$$H = \{x \in \mathbb{R}^n : c^\mathsf{T}x = \delta\} \quad \textit{mit} \quad c = z - y,\ \delta = c^\mathsf{T}y,\ y = \pi_C(z)$$

enthält y und ist somit eine Stützhyperebene von C. Eine strikte Trennhyperebene H von $\{z\}$ und C, also mit $z \in H^+$ und $z \notin H$, $C \subseteq H^-$ und $C \cap H = \emptyset$, erhält man mit $\delta \in (c^\mathsf{T}y, c^\mathsf{T}z)$.

KOROLLAR 3.9. *Sei $C \subseteq \mathbb{R}^n$ abgeschlossen, konvex, und $C \neq \emptyset$. Die metrische Projektion π_C ist Lipschitzstetig mit Lipschitzkonstante 1.*

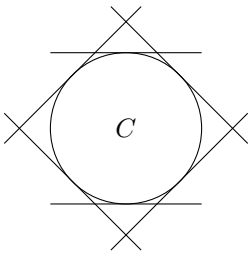
SATZ 3.10. *Sei $C \subseteq \mathbb{R}^n$, $C \neq \emptyset$ eine abgeschlossene und konvexe Menge. Dann gibt es für jeden Randpunkt $y \in \partial C$ von C einen Punkt $x \notin C$ mit $\pi_C(x) = y$.*

KOROLLAR 3.11. *Sei $C \subseteq \mathbb{R}^n$ eine abgeschlossene und konvexe Menge, die nicht leer ist. Sei $x \in \partial C$. Dann gibt es eine Stützhyperebene H von C mit $x \in H$.*

4. Repräsentation konvexer Mengen

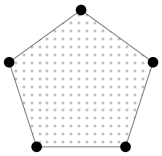
SATZ 4.1. *Sei $C \subseteq \mathbb{R}^n$ eine nichtleere, abgeschlossene und konvexe Menge. Dann gilt*

$$C = \bigcap_{H \text{ Stützhyperebene von } C} H^-$$

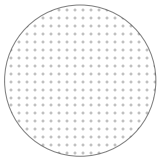


Die Charakterisierung aus Satz 4.1 liefert eine *äußere* Darstellung einer konvexen Menge, die z.B. verwendet werden kann, um zu überprüfen, ob ein Punkt in einer konvexen Menge liegt. Die Nachfolgende liefert eine *innere* Darstellung, die man etwa verwenden kann, um Punkte zu erzeugen, die innerhalb der Menge liegen.

DEFINITION 4.2. *Sei $C \subseteq \mathbb{R}^n$ eine konvexe Menge. Ein Punkt $z \in C$ heißt* Extrempunkt *von C, falls für alle $x, y \in C$ mit $z = \alpha x + (1-\alpha)y$ und $\alpha \in (0,1)$ stets $x = z = y$ folgt. Die Menge aller Extrempunkte von C wird mit* $\mathrm{ext}(C)$ *bezeichnet.*



5 Extrempunkte



∞- viele Extrempunkte

SATZ 4.3. *(Minkowski; Krein-Milman)*
Sei $C \subseteq \mathbb{R}^n$ eine kompakte und konvexe Menge. Dann gilt $C = \mathrm{conv}(\mathrm{ext}(C))$.

KAPITEL V

Polyedertheorie und Lineare Optimierung

1. Polyeder und Polytope

Bereits gesehen (Satz IV.4.1): Für allgemeine Mengen $C \subseteq \mathbb{R}^n$ nicht-leer, abgeschlossen, und konvex gilt:

$$C = \bigcap_{H \text{ Stützhyperebene von } C} H^-$$

Auch (Satz IV.4.3 von Krein-Milman und Minkowski): Für allgemeine kompakte und konvexe Mengen $C \subseteq \mathbb{R}^n$ gilt:

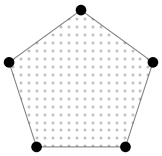
$$C = \operatorname{conv}(\operatorname{ext}(C))$$

Besonders wichtige Repräsentationen sind die, die eine endliche Beschreibung zulassen.

DEFINITION 1.1. *Eine Menge $P \subseteq \mathbb{R}^n$ heißt (konvexes)* Polyeder*, falls es eine Matrix $A \in \mathbb{R}^{m \times n}$ und einen Vektor $b \in \mathbb{R}^m$ gibt, so dass*

$$P = \{x \in \mathbb{R}^n : Ax \leq b\}$$

gilt. Die Extrempunkte von Polyedern heißen Ecken.



Ein Polyeder ist also darstellbar als Durchschnitt *endlich* vieler Halbräume. Ein Polyeder entspricht somit außerdem der Lösungsmenge eines Systems linearer Ungleichungen und somit der Menge der zulässigen Lösungen eines Linearen Programms.

DEFINITION 1.2. *Eine Menge $P \subseteq \mathbb{R}^n$ heißt (konvexes)* Polytop*, falls es eine endliche Menge $A = \{x_1, \ldots, x_N\}$ gibt mit*

$$P = \operatorname{conv}(A).$$

Das heißt, Polytope sind konvexe Hüllen *endlich* vieler Punkte.

SATZ 1.3. *Sei $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ein Polyeder und $z \in P$. Bezeiche mit A_z die Teilmatrix von A, die genau die Zeilen enthält mit $a_i^\intercal z = b_i$ („*aktive Ungleichungen an *z“). Der Vektor b_z bezeichne die zugehörigen Einträge von b. Dann gilt:*

$$z \text{ ist eine Ecke von } P \Leftrightarrow \operatorname{rang} A_z = n$$

KOROLLAR 1.4. *Ein Polyeder $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ mit $A \in \mathbb{R}^{m \times n}$ hat höchstens $\binom{m}{n}$ (also insbesondere endlich viele) Ecken.*

Wir studieren nun das Verhältnis zwischen Polyedern und Polytopen etwas genauer.

SATZ 1.5. *(Minkowski)*
Sei $P \subseteq \mathbb{R}^n$ ein beschränktes *Polyeder. Dann ist P die konvexe Hülle seiner endlich vielen Ecken.*
Mit anderen Worten: Ein beschränktes Polyeder ist ein Polytop.

Wir wollen nun auch die Umkehrung zeigen: Ein Polytop ist ein beschränktes Polyeder. Dazu benötigen wir noch ein Hilfsmittel.

DEFINITION 1.6. *Sei $A \subseteq \mathbb{R}^n$. Dann heißt*

$$A^\circ = \{y \in \mathbb{R}^n : x^\mathsf{T} y \leq 1 \ \text{ für alle } x \in A\}$$

die polare Menge *von A.*

LEMMA 1.7. *Es seien $A, B \subseteq \mathbb{R}^n$.*

(a) *Bezeichne mit αA die Menge $\{\alpha x : x \in A\}$ für ein $\alpha \in \mathbb{R}$. Dann gilt für $\alpha > 0$:*
$$(\alpha A)^\circ = \frac{1}{\alpha} A^\circ$$
(b) *Falls $A \subseteq B$, dann folgt $B^\circ \subseteq A^\circ$.*
(c) *$(B_n)^\circ = B_n$, wobei $B_n = B(0,1) = \{x \in \mathbb{R}^n : x^\mathsf{T} x \leq 1\}$ die n-dimensionale Einheitskugel ist.*
(d) *Falls $P = \operatorname{conv}\{x_1, \ldots, x_t\}$ ist, dann folgt*
$$P^\circ = \{y \in \mathbb{R}^n : x_1^\mathsf{T} y \leq 1, \ldots, x_t^\mathsf{T} y \leq 1\}.$$

SATZ 1.8. *(Weyl)*
Ein Polytop ist ein beschränktes Polyeder.

KOROLLAR 1.9. *(Theorem von Minkowski-Weyl)*
Sei $P \subseteq \mathbb{R}^n$. Dann gilt: P ist ein beschränktes Polyeder $\Leftrightarrow$ P ist ein Polytop.

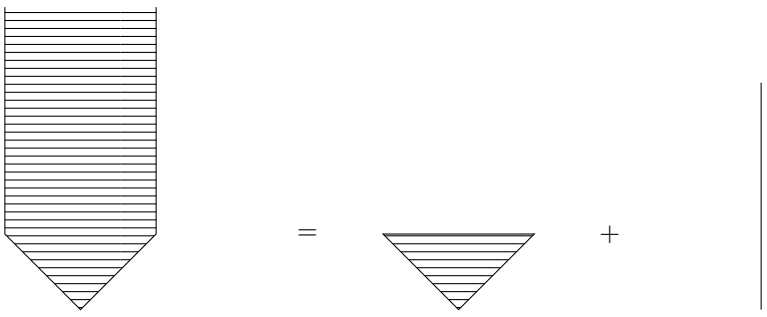
Zum Abschluss dieses Abschnitts betrachten wir noch zwei weitere klassische Theoreme von Minkowski und Weyl zur endlichen inneren Darstellung von (eventuell unbeschränkten) Polyedern. Für die zugehörigen Beweise wird z.B. Kapitel 7 des Lehrbuchs 'Theory of linear and integer programming' von A. Schrijver empfohlen.

SATZ 1.10. *(Minkowski-Weyl für Kegel)*
Ein konvexer Kegel $C \subseteq \mathbb{R}^n$ ist endlich erzeugt genau dann, wenn es eine Matrix $A \in \mathbb{R}^{m \times n}$ gibt mit
$$C = \{x \in \mathbb{R}^n : Ax \leq 0\}.$$

BEMERKUNG 1.11. *Insbesondere ist ein endlich erzeugter Kegel also ein Polyeder, und somit eine abgeschlossene und konvexe Menge.*

SATZ 1.12. *(Minkowski-Weyl, allgemeine Form)*
Eine Menge $P \subseteq \mathbb{R}^n$ ist ein Polyeder, genau dann, wenn es Vektoren $x_1, \ldots, x_s \in \mathbb{R}^n$ und $y_1, \ldots, y_t \in \mathbb{R}^n$ gibt, so dass
$$P = \operatorname{conv}\{x_1, \ldots, x_s\} + \operatorname{cone}\{y_1, \ldots, y_t\}$$
gilt. Dabei bezeichnet
$$A + B = \{a + b : a \in A, b \in B\}$$
die Minkowski-Summe von zwei Mengen $A, B \subseteq \mathbb{R}^n$.

BEISPIEL 1.13.



2. Das Lemma von Farkas

In diesem Abschnitt leiten wir ein Lösbarkeitskriterium für ein System von linearen Ungleichungen her, das uns auch auch den Weg zur Dualisierung linearer Programme (im nachfolgenden Abschnitt) weist.
Vorab jedoch betrachten wir zur Einstimmung ein ähnliches Lösbarkeitskriterium für lineare *Gleichungs*systeme.

SATZ 2.1. *Sei $Ax = b$ mit $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ ein lineares Gleichungssystem. Dann gilt:*

$$\exists x \in \mathbb{R}^n : Ax = b \Leftrightarrow \nexists y \in \mathbb{R}^m : y^\mathsf{T} A = 0 \text{ und } y^\mathsf{T} b \neq 0$$

SATZ 2.2. *(Farkas Lemma)*
Seien $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$ gegeben. Dann gilt:

$$\exists x \geq 0 : Ax = b \Leftrightarrow \nexists y \in \mathbb{R}^m : y^\mathsf{T} A \geq 0 \text{ und } y^\mathsf{T} b < 0$$

Das Lemma von Farkas ist ein Alternativsatz:
Entweder $\exists x \geq 0 : Ax = b$ oder $\exists y : y^\mathsf{T} A \geq 0,\ y^\mathsf{T} b < 0$.

KOROLLAR 2.3. *(Variante von Farkas Lemma)*
Seien $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$ gegeben. Dann gilt:

$$\exists x \in \mathbb{R}^n : Ax \leq b \Leftrightarrow \nexists y \geq 0 : y^\mathsf{T} A = 0,\ y^\mathsf{T} b < 0$$

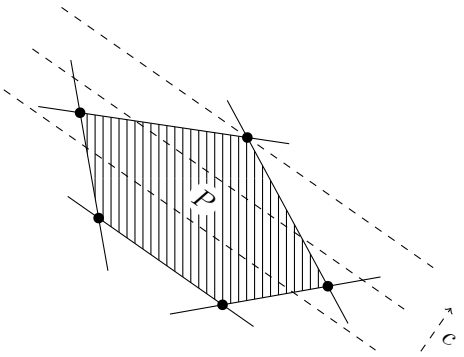
3. Lineare Optimierung / Lineare Programmierung / Dualität

DEFINITION 3.1. *Ein* lineares Programm *(LP) in primaler Standardform ist ein Maximierungsproblem von der Form*

$$\text{(LP)} \qquad \begin{aligned} &\sup\ c^\mathsf{T} x \\ &Ax \leq b, \\ &x \in \mathbb{R}^n \end{aligned}$$

wobei $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ gegeben sind.

Geometrische Interpretation von (LP): Maximiere die lineare Funktion $x \mapsto c^\mathsf{T} x$ über dem Polyeder $P = \{x \in \mathbb{R}^n : Ax \leq b\}$. Das heißt: Verschiebe eine zu c orthogonale Hyperebene so weit in Richtung c, dass ihr Schnitt mit P gerade noch nicht leer ist.



Sprechweise:
x heißt *gültige/zulässige Lösung* $\Leftrightarrow x \in P$.
x heißt *optimale Lösung* $\Leftrightarrow x \in P$ und $c^\mathsf{T} x = \max\{c^\mathsf{T} y : y \in P\}$.
(LP) heißt *unbeschränkt* $\Leftrightarrow \sup\{c^\mathsf{T} x : x \in P\} = +\infty$.
(LP) heißt *ungültig/unzulässig* $\Leftrightarrow \nexists x \in P$ (dann ist $\sup\{c^\mathsf{T} x : x \in P\} = -\infty$).

Satz 3.2. *Sei mit dem Polyeder $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ die Menge der zulässigen Lösungen eines linearen Programms $\max\{c^\mathsf{T} x : x \in P\}$ gegeben. Falls P nicht leer und beschränkt ist, so gibt es eine Ecke z von P, die eine optimale Lösung des linearen Programms ist, d.h. $c^\mathsf{T} z \geq c^\mathsf{T} x$ für alle $x \in P$.*

Daraus ergibt sich ein offensichtlicher (aber i.A. sehr langsamer) Algorithmus zur Lösung von (LP), falls P beschränkt ist: Bestimme alle Ecken $x_1, \ldots, x_t$ von P und finde einen Index i_0 mit $c^\mathsf{T} x_{i_0} \geq c^\mathsf{T} x_i \; \forall i = 1, \ldots, t$.

Problem: Die Anzahl der Ecken kann sehr groß werden. Eine Beschreibung des n-dimensionalen Würfels

$$\{x \in \mathbb{R}^n : -1 \leq x_i \leq 1, i = 1, \ldots, n\}$$

hat zum Beispiel nur $2n$ Ungleichungen, der Würfel aber 2^n Ecken.

Bessere Algorithmen lernen wir im nächsten Kapitel kennen. In diesem Kapitel beschäftigen wir uns zunächst mit Optimalitätsbedingungen und Dualitätstheorie.

Definition 3.3. *Seien $A \in \mathbb{R}^{m \times n}, c \in \mathbb{R}^n, b \in \mathbb{R}^m$ und das zugehörige (LP) in Standardform gegeben. Wir nennen dieses Maximierungsproblem nun das* primale LP *(PLP):*

$$\text{(PLP)} \qquad \begin{aligned} \sup \; & c^\mathsf{T} x \\ & Ax \leq b, \\ & x \in \mathbb{R}^n \end{aligned}$$

Das zu (PLP) zugehörige duale *lineare Programm in (dualer) Standardform ist ein Minimierungsproblem von der Form*

$$\text{(DLP)} \qquad \begin{aligned} \inf \; & b^\mathsf{T} y \\ & A^\mathsf{T} y = c \\ & y \geq 0 \\ & y \in \mathbb{R}^m \end{aligned}$$

Satz 3.4. *(*schwache Dualität*)*
Sei $\tilde{x}$ zulässig für (PLP) und $\tilde{y}$ zulässig für (DLP), dann ist

$$c^\mathsf{T} \tilde{x} \leq b^\mathsf{T} \tilde{y}.$$

Die Beziehung gilt also insbesondere für ein Paar optimaler *Lösungen x^* und y^* für (PLP) und (DLP).*

Theorem 3.5. *(*starke Dualität*, von Neumann (1947))*
Falls (PLP) und (DLP) beide zulässige Lösungen besitzen, dann besitzen beide auch optimale Lösungen x^*, y^* für die gilt:

$$p^* := c^\mathsf{T} x^* = \max\{c^\mathsf{T} x : Ax \leq b\} = \min\{b^\mathsf{T} y : A^\mathsf{T} y = c, y \geq 0\} = b^\mathsf{T} y^* =: d^*$$

Korollar 3.6. *Es gilt*

$$\max\{c^\mathsf{T} x : x \geq 0, \; Ax = b\} = \min\{b^\mathsf{T} y : A^\mathsf{T} y \geq c\},$$

falls beide Mengen gültiger Lösungen nicht leer sind.

SATZ 3.7. *(*Optimalitätsbedingung*)*
Seien $\tilde{x}, \tilde{y}$ zulässig für (PLP) bzw. (DLP). Dann ist $\tilde{x}$ optimal für (PLP) und $\tilde{y}$ optimal für (DLP) genau dann, wenn $\tilde{y}^\mathsf{T}(A\tilde{x} - b) = 0$.

KOROLLAR 3.8. *(*Komplementarität*)*
Seien $\tilde{x}, \tilde{y}$ zulässig (und optimal) für (PLP) bzw. (DLP). Dann gelten die folgenden Komplementaritätsbedingungen *für alle $j = 1, \ldots, m$:*

$$\tilde{y}_j \neq 0 \Rightarrow (A\tilde{x} - b)_j = 0$$
$$(A\tilde{x} - b)_j \neq 0 \Rightarrow \tilde{y}_j = 0.$$

BEMERKUNG 3.9. *Aus dem schwachen Dualitätssatz folgt:*

- *Wenn (PLP) unbeschränkt ist, so ist (DLP) unzulässig.*
- *Wenn (DLP) unbeschränkt ist, so ist (PLP) unzulässig.*

Die möglichen Kombinationen sind:

		(DLP)		
		optimal	*unzulässig*	*unbeschränkt*
(PLP)	*optimal*	×		
	unzulässig		×	×
	unbeschränkt		×	

KAPITEL VI

Algorithmen der linearen Optimierung

Ein fundamentales Problem in der linearen Algebra ist, zu entscheiden, ob die Lösungsmenge eines linearen Gleichungssystems $\{x \in \mathbb{R}^n : Ax = b\}$ nicht leer ist, und der wohl wichtigste Algorithmus dazu ist das Gaußsche Eliminationsverfahren, mit dem im positiven Fall auch eine entsprechende Lösung berechnet werden kann.

Ein Grundproblem in der Polyedertheorie ist, zu entscheiden, ob ein Polyeder $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ nicht leer ist. Das nachfolgend vorgestellte Verfahren von Fourier und Motzkin ist eine Variante des Gaußschen Eliminationsverfahrens, mit dem man diese Frage beantworten, und (theoretisch) auch LPs lösen, kann.

1. Das Eliminationsverfahren von Fourier und Motzkin

Gegeben seien $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$. Ziel ist es, $x \in \mathbb{R}^n$ mit $Ax \leq b$ zu finden bzw. zu entscheiden (mit mathematischer Sicherheit), dass es ein solches x nicht gibt.
Idee: Eliminiere die Variable x_1. Finde $\tilde{A} \in \mathbb{R}^{\tilde{m} \times (n-1)}$, $\tilde{b} \in \mathbb{R}^{\tilde{m}}$, so dass

$$\exists x \in \mathbb{R}^n : Ax \leq b \Leftrightarrow \exists \tilde{x} \in \mathbb{R}^{n-1} : \tilde{A}\tilde{x} \leq \tilde{b}.$$

Dazu multiplizieren wird die Zeilen von A und die entsprechenden Einträge von b mit *positiven* Konstanten, so dass die erste Spalte des Systems nur noch Einträge 0, 1, -1 hat (zur Übersichtlichkeit). Dann hat das System $Ax \leq b$ nach Umnummerierung der Zeilen folgende Gestalt:

$$(*) \qquad \begin{bmatrix} 1 & (a_1')^\mathsf{T} \\ \vdots & \vdots \\ 1 & (a_r')^\mathsf{T} \\ -1 & (a_{r+1}')^\mathsf{T} \\ \vdots & \vdots \\ -1 & (a_{r+s}')^\mathsf{T} \\ 0 & (a_{r+s+1}')^\mathsf{T} \\ \vdots & \vdots \\ 0 & (a_m')^\mathsf{T} \end{bmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \leq \begin{pmatrix} b_1' \\ \vdots \\ b_m' \end{pmatrix},$$

wobei $(a_i')^\mathsf{T} \in \mathbb{R}^{n-1}$ die i-te Zeile von A ist, in der das erste Element gelöscht wurde (es kann passieren, dass $r = 0$ oder $s = 0$ ist). Betrachte die ersten r Bedingungen:

$$x_1 + (a_i')^\mathsf{T} \underbrace{\begin{pmatrix} x_2 \\ \vdots \\ x_n \end{pmatrix}}_{=\tilde{x}} \leq b_i' \Longleftrightarrow x_1 \leq b_i' - (a_i')^\mathsf{T}\tilde{x}, \quad i = 1, \ldots, r.$$

Genauso die nächsten s Bedingungen:

$$-x_1 + (a_{r+j}')^\mathsf{T}\tilde{x} \leq b_{r+j}' \Longleftrightarrow x_1 \geq (a_{r+j}')^\mathsf{T}\tilde{x} - b_{r+j}', \quad j = 1, \ldots, s.$$

Zusammen gilt also

$$(**) \qquad \sup_{j=1,\dots,s} (a'_{r+j})^{\mathsf{T}}\tilde{x} - b'_{r+j} \leq x_1 \leq \inf_{i=1,\dots,r} b'_i - (a'_i)^{\mathsf{T}}\tilde{x}.$$

Falls $s = 0$, dann ist $\sup_{j=1,\dots,s}(a'_{r+j})^{\mathsf{T}}\tilde{x} - b'_{r+j} = -\infty$ und falls $r = 0$, ist $\inf_{i=1,\dots,r} b'_i - (a'_i)^{\mathsf{T}}\tilde{x} = +\infty$. In diesen Fällen ist also P in die jeweilige Richtung bzgl. x_1 unbeschränkt.

Nun kann man x_1 eliminieren und das System $(*)$ ist genau dann lösbar, wenn das System

$$\begin{aligned} &(a'_{r+j})^{\mathsf{T}}\tilde{x} - b'_{r+j} \leq b'_i - (a'_i)^{\mathsf{T}}\tilde{x}, \quad i = 1,\dots,r \text{ und } j = 1,\dots,s \\ &(a'_i)^{\mathsf{T}}\tilde{x} \leq b'_i, \quad i = r+s+1,\dots,m \end{aligned}$$

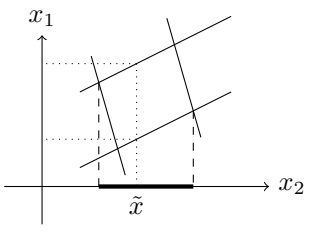
bzw. das System

$$(***) \qquad \begin{aligned} &((a'_{r+j})^{\mathsf{T}} + (a'_i)^{\mathsf{T}})\tilde{x} \leq b'_i + b'_{r+j}, \quad i = 1,\dots,r \text{ und } j = 1,\dots,s \\ &(a'_i)^{\mathsf{T}}\tilde{x} \leq b'_i, \quad i = r+s+1,\dots,m. \end{aligned}$$

lösbar ist. Das neue System $\tilde{A}\tilde{x} \leq \tilde{b}$ hat $\tilde{m} = r \cdot s + m - (r+s)$ viele Ungleichungen und $n-1$ Variablen.

Bemerkung 1.1.

(1) $(***)$ *entspricht der Projektion des Polyeders* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *entlang der* x_1*-Achse (auf die* x_2 *Achse).*



(2) *Eine Lösung* $\tilde{x}$ *kann zu einer Lösung* $(x_1, \tilde{x})$ *von* $(*)$ *erweitert werden. Dazu muss* x_1 *die Ungleichungen* $(**)$ *erfüllen.*
(3) *Das Verfahren wird fortgesetzt, indem nun sukzessive die Variablen* $x_2, x_3, \dots$ *eliminiert werden, bis man bei* x_n *angekommen ist.*
(4) *Für* x_n *ist es offensichtlich, ob das finale System eine Lösung besitzt. Das finale System hat genau dann eine Lösung, wenn das Ursprungssystem* $(*)$ *eine Lösung besitzt.*

ALGORITHMUS 1.2. *„Fourier-Motzkin Elimination“*

Eingabe : $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$
Ausgabe : $x \in \mathbb{R}^n$ *mit* $Ax \leq b$ *oder „*$Ax \leq b$ *hat keine Lösung“.*
$A_0 = A$, $b_0 = b$.
for $k = 0, \ldots, n-2$ **do**
  Bringe das System $\begin{bmatrix} A_k & b_k \end{bmatrix}$ *in die Form* $(*)$.
  Definiere $\begin{bmatrix} A_{k+1} & b_{k+1} \end{bmatrix}$ *entsprechend* $(***)$.
end
if $A_{n-1}x_n \leq b_{n-1}$ *lösbar* **then**
  Wähle x_n *fest.*
  for $k = n-1, \ldots, 1$ **do**
    Wähle x_k *fest, so dass* $(**)$ *erfüllt ist (mit* x_k *statt* x_1 *gelesen).*
  end
  Gib $x = (x_1, \ldots, x_n)^T$ *zurück.*
else
  „$Ax \leq b$ *hat keine Lösung“.*
end

Das Verfahren kann auch verwendet werden, um ein LP

$$\sup\{c^\mathsf{T}x : Ax \leq b,\ x \in \mathbb{R}^n\},$$

zu lösen. Dazu führen wir eine zusätzliche Variable λ ein und betrachten das System

$$Ax \leq b,\ \lambda \leq c^\mathsf{T}x.$$

Wegen $\lambda \leq c^\mathsf{T}x \iff \lambda - c^\mathsf{T}x \leq 0$, ist das System äquivalent zu

$$\begin{bmatrix} A & 0 \\ -c^\mathsf{T} & 1 \end{bmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} \leq \begin{pmatrix} b \\ 0 \end{pmatrix}.$$

Man kann nun das LP lösen, indem man eine Lösung dieses Systems findet, bei der λ maximal ist. Dazu eliminiert man $x_1, \ldots, x_n$ bis λ die letzte Variable ist. Dann wählt man λ so groß wie möglich. Ebenso kann eine mögliche Unbeschränktheit von (LP) erkannt werden. Diese liegt vor, falls λ nach oben unbeschränkt ist, bzw. zuvor bereits ein x_i, $i \in \{1, \ldots, n\}$ auftritt, so dass

$$c_i > 0 \text{ und } r = 0 \text{ in } (**), \text{ oder}$$
$$c_i < 0 \text{ und } s = 0 \text{ in } (**).$$

In der Praxis verwendet man das Eliminationsverfahren von Fourier und Motzkin jedoch nicht um LPs zu lösen. Den Grund dafür liefert der nachfolgende Satz.

SATZ 1.3. *Das bei der Fourier-Motzkin Elimination zu betrachtende Ungleichungssystem* $A_{n-1}x_n \leq b_{n-1}$ *(und damit auch die Anzahl der benötigten Rechenschritte) ist im schlimmsten Fall von der Größenordnung* m^{2^n}.

Dennoch wird die Methode (für sehr kleine n) in der Praxis eingesetzt, etwa um minimale äußere lineare Beschreibungen von Polytopen zu berechnen, von denen a priori nur eine innere Beschreibung bekannt ist (oder umgekehrt), vgl. z.B. wissenschaftliche Softwarepakete wie „PORTA“, „PANDA“, oder „Polymake“.

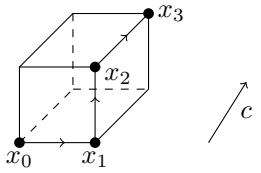
2. Das Simplexverfahren

Ziel: Löse (LP)

$$\begin{aligned} p^* = \sup\ & c^\mathsf{T} x \\ & Ax \leq b \\ & x \in \mathbb{R}^n. \end{aligned}$$

Wissen: Falls $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ in Richtung c beschränkt ist, so wird das Maximum an einer Ecke von P angenommen.
Geometrische Idee: Finde eine Folge von Ecken $x_0, x_1, \ldots, x_N \in P$, so dass

$$c^\mathsf{T} x_0 \leq c^\mathsf{T} x_1 \leq \ldots \leq c^\mathsf{T} x_N = p^*.$$



Das Simplexverfahren wurde 1947 von dem US-amerikanischen Mathematiker George Dantzig entwickelt. Noch heute ist es eines der wichtigsten und vielleicht sogar das meistgenutzte algorithmische Verfahren in der mathematischen Optimierung.

3. Simplexalgorithmus mit bekannter Startecke

Zunächst treffen wir die Annahme, dass das Polyeder P eine Ecke x_0 besitzt, die wir kennen. Später klären wir, wie wir diese Ecke finden.

ALGORITHMUS 3.1. *Simplexalgorithmus*

Eingabe : $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n, x_0$ *Ecke von* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$
Ausgabe : *Ecke* x_N *von* P *mit* $p^* = c^T x_N$, *oder* $p^* = \infty$.
Wähle $n \times n$*-Teilsystem* $A_0 x \leq b_0$ *v.* $Ax \leq b$ *mit* $A_0 x_0 = b_0$ *u.* $\operatorname{rang}(A_0) = n$.
Setze $k = 0$.
Wiederhole:
Bestimme $u_k \in \mathbb{R}^m$ *mit* $c = A^\mathsf{T} u_k$.
Setze $(u_k)_i = 0$, *falls Zeile* i *von* A *nicht zu* A_k *gehört.*
if $u_k \geq 0$ **then**
  Gebe x_k *zurück. STOP.* $(*)$
else
  Sei i *der kleinste Index (bezogen auf die Zeilen von* A_k*) mit* $(u_k)_i < 0$.
  Bestimme $d \in \mathbb{R}^n$ *mit* $d = -(A_k^{-1})_{\cdot i}$ *durch Lösen von* $A_k d = -e_i$.
  D.h. $a_j^\mathsf{T} d = 0$ *für alle Zeilen* a_j^T, $j \neq i$, *von* A_k, *und* $a_i^\mathsf{T} d = -1$.
  if $a_j^\mathsf{T} d \leq 0$ *für alle Zeilen* a_j^T *von* A **then**
    Gebe $p^* = \infty$ *zurück. STOP.* $(**)$
  else
    Setze $\lambda_k = \min\{\frac{b_j - a_j^\mathsf{T} x_k}{a_j^\mathsf{T} d} : j = 1, \ldots, m,\ a_j^\mathsf{T} d > 0\}$. $(***)$
    Sei j *der kleinste Index, an dem das Minimum angenommen wird.*
    Setze:
    $A_{k+1} = A_k$ *mit* a_i^T *durch* a_j^T *ersetzt.*
    $x_{k+1} = x_k + \lambda_k d$
    $b_{k+1} = A_{k+1} x_{k+1}$
    $k = k + 1$
  end
end

Erläuterungen zur Korrektheit:

zu $(*)$: Falls $u_k \geq 0$, so ist x_k optimal, denn u_k ist eine optimale Lösung des dualen LPs:

$$c^\mathsf{T} x_k = (A^\mathsf{T} u_k)^\mathsf{T} x_k = u_k^\mathsf{T} A x_k \underbrace{=}_{\substack{(u_k)_i = 0 \text{ für } a_i^T \\ \text{nicht in } A_k}} u_k^\mathsf{T} b \geq \min\{b^\mathsf{T} y : y \geq 0, A^\mathsf{T} y = c\}$$
$$= \max\{c^\mathsf{T} x : Ax \leq b\}.$$

zu $(**)$: Falls $a_j^\mathsf{T} d \leq 0$ für alle Zeilen a_j^T von A, dann ist $p^* = \infty$. Denn $Ax \leq b$ und $Ad \leq 0 \Rightarrow A(x+d) \leq b$, und somit $x_k + \lambda d \in P$ für $\lambda \geq 0$. Außerdem gilt:

$$\begin{aligned} c^\mathsf{T}(x_k + \lambda d) &= c^\mathsf{T} x_k + \lambda c^\mathsf{T} d \\ &= c^\mathsf{T} x_k + \lambda \underbrace{u_k^\mathsf{T} A d}_{=-(u_k)_i} \\ &= c^\mathsf{T} x_k - \lambda \underbrace{(u_k)_i}_{<0} \to \infty \text{ für } \lambda \to \infty \end{aligned}$$

zu $(***)$ und der Bestimmung von λ_k:

$$\begin{aligned} \lambda_k &= \max\{\lambda : x_k + \lambda d \in P\} \\ &= \min\left\{\frac{b_j - a_j^\mathsf{T} x_k}{a_j^\mathsf{T} d} : j = 1, \ldots, m,\ a_j^\mathsf{T} d > 0\right\}, \end{aligned}$$

denn:

$$\begin{aligned} x_k + \lambda d \in P &\Leftrightarrow A(x_k + \lambda d)_j \leq b_j,\ j = 1, \ldots, m \\ &\Leftrightarrow a_j^\mathsf{T} x_k + \lambda a_j^\mathsf{T} d \leq b_j,\ j = 1, \ldots, m \\ &\Leftrightarrow \lambda \leq \frac{b_j - a_j^\mathsf{T} x_k}{a_j^\mathsf{T} d},\ j = 1, \ldots, m \end{aligned}$$

BEMERKUNG 3.2. *Die Vorschrift (sog. „Pivotregel"), austretende und eintretende Zeile jeweils mit kleinstmöglichem Index zu wählen, geht zurück auf Robert G. Bland („Bland's Rule", 1977).*

SATZ 3.3. *Mit Blands Regel terminiert der Simplexalgorithmus.*

BEMERKUNG 3.4. *Seien $A \in \mathbb{R}^{m \times n}$ und $b \in \mathbb{R}^m$, sowie $L = \max\{|a_{ij}|, |b_i|\}$, $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$. Bisher ist keine Pivotregel bekannt, unter deren Verwendung die Anzahl der besuchten Ecken stets durch ein Polynom in n, m, und $\log_2 L$ beschränkt ist. Umgekehrt existiert für die meisten Regeln ein (künstliches) Beispiel mit einer exponentiellen Anzahl besuchter Ecken. Wird ein LP jedoch nur leicht zufällig verändert („perturbiert"), kann eine polynomielle erwartete Laufzeit gezeigt werden (Spielman-Teng (2004), „smoothed analysis"). In der Praxis ist der Simplexalgorithmus sehr schnell, und es existieren u.a. hochperformante kommerzielle Implementierungen (etwa „CPLEX", und „GuRoBi"). Gründe für seine Verwendung (neben der praktischen Geschwindigkeit) sind z.B.: Besonders effiziente Verwaltung unterer und oberer Schranken für Variablen. „Warmstarten" nach Hinzufügen von Ungleichungen (duale Zulässigkeit bleibt erhalten / kann leicht hergestellt werden) oder Variablen (primale Zulässigkeit bleibt erhalten).*

4. Simplexalgorithmus ohne Startecke

Abschließend beschäftigen wir uns mit der Frage, wie man den Simplexalgorithmus startet, wenn man keine Ecke x_0 kennt.

OBdA: Das LP ist von der Form $\sup\{c^\mathsf{T} x : x \geq 0, Ax \leq b\}$.

Idee: Um eine Ecke von $P = \{x \in \mathbb{R}^n : x \geq 0, Ax \leq b\}$ zu finden, füge zusätzliche Variablen hinzu und stelle ein neues LP auf, das eine offensichtliche Ecke besitzt und dessen optimale Lösung eine Ecke von P liefert, sofern eine solche existiert.

Zusatzvariablen: $y \in \mathbb{R}^m,\ y \geq 0$.

Neues LP:

$$\min 1^\mathsf{T} y$$
$$\begin{bmatrix} A & -I_m \\ -I_n & 0 \\ 0 & -I_m \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix}$$

mit $1 = (1, \ldots, 1)^\mathsf{T}$.

Offensichtliche Ecke:

$$\tilde{x} = 0, \quad \tilde{y}_j = \begin{cases} 0, & \text{falls } b_j \geq 0 \\ -b_j, & \text{falls } b_j < 0 \end{cases},\ j = 1, \ldots, m.$$

Dann ist $\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \in \mathbb{R}^{n+m}$ eine Ecke von $P' = \{\begin{pmatrix} x \\ y \end{pmatrix} : Ax - y \leq b, x \geq 0, y \geq 0\}$, weil $\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \in P'$ und rang $\begin{bmatrix} A & -I_m \\ -I_n & 0 \\ 0 & -I_m \end{bmatrix}_{\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}} = n + m.$

Jetzt kann man den Simplexalgorithmus mit der Startecke $\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}$ verwenden, um das neue LP zu lösen. Sei $\begin{pmatrix} x^* \\ y^* \end{pmatrix}$ die Ecke von P', die der Algorithmus liefert.

1.Fall: $1^\mathsf{T} y^* > 0$.
Dann gilt:

$$\exists j : y_j^* > 0 \text{ und } (Ax - y^*)_j \leq b_j.$$

Da wir minimieren, bedeutet dies

$$\nexists x \geq 0 : Ax \leq b,$$

also hat das ursprüngliche LP keine Lösung (ist unzulässig).

2.Fall: $1^\mathsf{T} y^* = 0$. Dann ist $y^* = 0$, und x^* eine Ecke von P, weil $x^* \in P$ und rang $\begin{bmatrix} A \\ -I \end{bmatrix}_{x^*} = n$.

Bemerkung 4.1. *Zur Komplexität der Linearen Optimierung:*
Bis heute ist offen, ob

(1) *der maximale Abstand zwischen zwei Ecken polynomiell in n, m ist („polynomielle Hirsch-Vermutung“).*
(2) *ein* streng-*polynomieller Algorithmus (d.h. mit Laufzeit polynomiell abhängig nur von n, m) zur Lösung linearer Programme existiert.*

5. Ellipsoidmethode: Grundlagen

DEFINITION 5.1. *Eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt* positiv definit, *falls $x^\mathsf{T} Ax > 0$ für alle $x \in \mathbb{R}^n \setminus \{0\}$. Ist A symmetrisch, so ist A positiv definit genau dann, wenn alle Eigenwerte $\lambda_1, \dots, \lambda_n$ von A positiv sind.*

BEMERKUNG 5.2. *Wenn die Matrix $A \in \mathbb{R}^{n\times n}$ symmetrisch und positiv definit ist, so existiert eine* Spektralzerlegung

$$A = \sum_{i=1}^{n} \lambda_i u_i u_i^\mathsf{T},$$

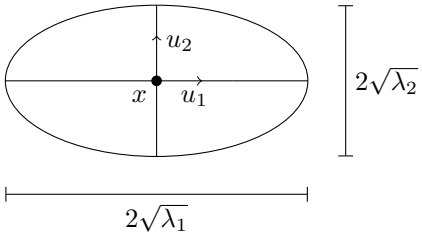
wobei $\lambda_1, \dots, \lambda_n > 0$ die Eigenwerte von A sind, und $u_1, \dots, u_n \in \mathbb{R}^n$ eine Orthonormalbasis bilden, bestehend aus den zugehörigen Eigenvektoren.

DEFINITION 5.3. *Eine positiv definite Matrix $A \in \mathbb{R}^{n\times n}$ und ein Vektor $x \in \mathbb{R}^n$ definieren das* Ellipsoid

$$\mathcal{E}(A, x) = \{y \in \mathbb{R}^n : (y - x)^\mathsf{T} A^{-1}(y - x) \le 1\}$$

mit „Mittelpunkt“ x.

BEMERKUNG 5.4. *Die Vektoren u_i der Spektralzerlegung bestimmen die Richtungen der Hauptachsen des Ellipsoids. Die Länge der jeweiligen Hauptachse ist durch $2\sqrt{\lambda_i}$ gegeben.*



BEISPIEL 5.5. *Für ein $r \in \mathbb{R}$ ist*

$$\begin{aligned}\mathcal{E}(r^2 I_n, 0) &= \{y \in \mathbb{R}^n : y^\mathsf{T} \frac{1}{r^2} I_n y \le 1\} \\ &= \{y \in \mathbb{R}^n : \frac{1}{r^2}\|y\|^2 \le 1\} \\ &= \{y \in \mathbb{R}^n : \|y\| \le r\} \\ &= r \cdot B_n\end{aligned}$$

wobei $B_n = \{y \in \mathbb{R} : \|y\| \le 1\}$ die n-dimensionale Einheitskugel ist.

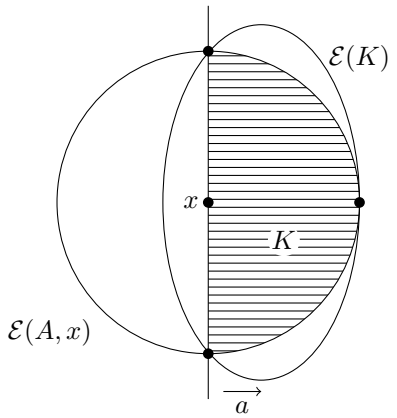
SATZ 5.6. *Das Volumen von $\mathcal{E}(A, x)$ ist*

$$\operatorname{vol} \mathcal{E}(A, x) = \sqrt{det\ A} \cdot \operatorname{vol} B_n,$$

wobei $\operatorname{vol} B_n = \frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2}+1)}$, *mit der Eulerschen Gammafunktion Γ, die für unsere Zwecke ausreichend definiert ist durch*

$$\Gamma(1/2) = \sqrt{\pi}, \quad \Gamma(1) = 1, \quad \Gamma(x + 1) = x \cdot \Gamma(x).$$

SATZ 5.7. *Sei $A \in \mathbb{R}^{n\times n}$ positiv definit, $x \in \mathbb{R}^n$, und $d \in \mathbb{R}^n \setminus \{0\}$.*
Betrachte die Menge $K = \mathcal{E}(A, x) \cap \{y \in \mathbb{R}^n : d^\mathsf{T} y \ge d^\mathsf{T} x\}$.



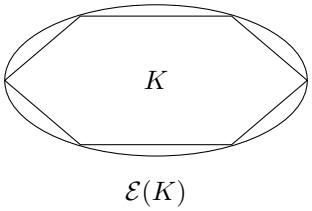
Dann existiert ein eindeutiges Ellipsoid $\mathcal{E}(K) \supseteq K$ mit minimalem Volumen. $\mathcal{E}(K)$ heißt (das) Loewner-John-Ellipsoid *(von K). Es ist $\mathcal{E}(K) = \mathcal{E}(A', x')$ mit*

$$A' = \frac{n^2}{n^2-1}(A - \frac{2}{n+1}bb^\mathsf{T}), \text{ und } x' = x + \frac{1}{n}b, \text{ wobei } b = \frac{1}{\sqrt{d^\mathsf{T}Ad}}Ad.$$

Außerdem ist

$$\frac{\operatorname{vol}\mathcal{E}(A',x')}{\operatorname{vol}\mathcal{E}(A,x)} < e^{-\frac{1}{2(n+1)}} < 1.$$

Bemerkung 5.8. *Das* Loewner-John-Ellipsoid *existiert für jede konvexe und kompakte Menge $K \subseteq \mathbb{R}^n$, $K \neq \emptyset$, und bildet die beste äußere ellipsoide Approximation von K.*



6. Ellipsoidmethode: Trennen und Optimieren

Sei $\mathcal{K}$ eine Klasse konvexer und kompakter Mengen $K \in \mathcal{K}$, $K \subseteq \mathbb{R}^n$. Wir betrachten folgende bereits behandelte algorithmische Problemstellungen in allgemeinerer Form:

Definition 6.1. *Trennungsproblem (Separationsproblem)*
Eingabe: $x \in \mathbb{R}^n$, $K \in \mathcal{K}$
Ausgabe: $x \in K$ *oder* $d \in \mathbb{R}^n$ *mit* $d^\mathsf{T}x > \max_{y\in K} d^\mathsf{T}y$.

D.h. falls $x \notin K$ soll eine Trennhyperebene $H = \{y \in \mathbb{R}^n : d^\mathsf{T}y = \delta\}$ von x und K zurückgegeben werden. Hierbei kann $\delta = d^\mathsf{T}x$ gewählt werden.

Definition 6.2. *Zulässigkeitsproblem*
Eingabe: $K \in \mathcal{K}$
Ausgabe: $x \in K$ *oder „*$K = \emptyset$*“.*

Definition 6.3. *Optimierungsproblem*
Eingabe: $c \in \mathbb{R}^n$, $\|c\| = 1$, $\varepsilon > 0$, $K \in \mathcal{K}$.
Ausgabe: $x \in K$ *mit* $c^\mathsf{T}x \geq \max_{y\in K} c^\mathsf{T}y - \varepsilon$.

BEISPIEL 6.4. *Falls* $K = \{x \in \mathbb{R}^n : Ax \leq b\}$ *ein (beschränktes) Polyeder ist, kann das Trennungsproblem für ein gegebenes* x^* *durch Einsetzen in alle linearen Ungleichungen*

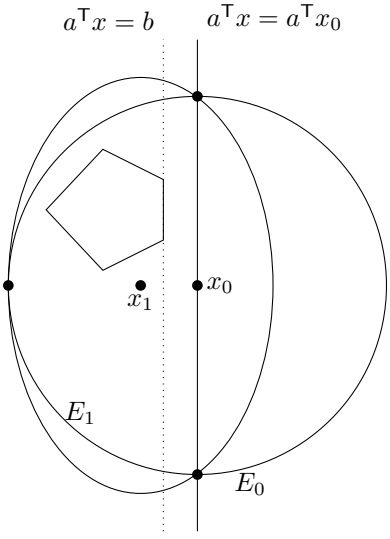
$$a_i^\mathsf{T} x \leq b_i \qquad i = 1, \ldots, m.$$

gelöst werden. Es gilt $x^* \in K$ *genau dann, wenn alle Ungleichungen erfüllt sind. Falls* $a_i^\mathsf{T} x > b_i$*, wähle* $d = a_i$*.*

SATZ 6.5. *(Khachiyan, 1979)*
Sei $K \in \mathcal{K}$*,* $K \subseteq \mathbb{R}^n$ *mit* $\dim K = n$*. Seien zudem* $x_0 \in \mathbb{R}^n$ *und* $R > 0$*, so dass* $K \subseteq x_0 + RB_n$*. Dann kann das Zulässigkeitsproblem für* K *durch eine (schwach-)polynomielle Anzahl von Trennungsproblemen gelöst werden.*

ALGORITHMUS 6.6. *Ellipsoidmethode (Khachiyan, 1979)*

Eingabe : $K \in \mathcal{K}$, $x_0 \in \mathbb{R}^n$, R *(mit* $x_0 + RB_n \supseteq K$*),* $N \in \mathbb{N}$
Ausgabe : $x \in K$ *oder „*$K = \emptyset$*“.*
$E_0 = \mathcal{E}(R^2 I_n, x_0)$
for $k = 0$ ***to*** $N - 1$ **do**
  Löse Trennungsproblem für x_k*, den Mittelpunkt von* E_k
  if $x_k \in K$ **then**
    Gebe x_k *als Lösung aus.*
  else
    Sei die Trennhyperebene definiert durch $a \in \mathbb{R}^n \setminus \{0\}$*.*
    Setze $d = -a$ *und*
    $E_{k+1} = \mathcal{E}(E_k \cap \{y \in \mathbb{R}^n : d^\mathsf{T} y \geq d^\mathsf{T} x_k\})$ *(Loewner-John-Ellipsoid)*
  end
end



Es ist insbesondere zu klären, wie N und R zu wählen sind, so dass ein korrektes und schwach-polynomielles Verfahren resultiert. Dazu ist noch etwas Vorarbeit nötig.

SATZ 6.7. *Sei* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *ein Polytop mit* $A \in \mathbb{Z}^{m \times n}$*,* $b \in \mathbb{Z}^m$*, und* $\dim P = n$*. Sei zudem* $L = \max\{|a_{ij}|, |b_i|\}$*,* $i \in \{1, \ldots, m\}$*,* $j \in \{1, \ldots, n\}$*. Dann gilt:*

(1) *Die Ecken von P liegen innerhalb der Menge*

$$\{x \in \mathbb{R}^n : -n^{\frac{n}{2}} L^n \leq x \leq n^{\frac{n}{2}} L^n\}.$$

(2) $\text{vol}(P) \geq \frac{1}{(nL)^{3n^2}}$.

Beweisskizze:

zu (1) Nach der Hadamard-Ungleichung gilt für jede $n \times n$-Teilmatrix A_z von A

$$|\det(A_z)| \leq \prod_{i=1}^{n} \|a_i\|,$$

wobei a_i die i-te Spalte von A_z bezeichnet. Nach Voraussetzung ist also $\|a_i\| \leq (nL^2)^{\frac{1}{2}}$ für $i = \{1, \ldots, n\}$ und somit auch $|det(A_z)| \leq n^{\frac{n}{2}} L^n$. Sei nun $z \in P$ eine Ecke von P. Dann ist nach der Cramerschen Regel $z_i = \frac{\det A_{z_i}}{\det A_z}$, wobei in A_{z_i} die i-te Spalte von A_z durch b ersetzt ist.

zu (2) Da P volldimensional ist, existieren $n+1$ affin-unabhängige Vektoren (Ecken) $x_0, \ldots x_n$ (von P), die einen n-dimensionalen Simplex im $\mathbb{R}^n$ aufspannen, und deren Komponenten ebenfalls wie oben beschränkt sind. Das Volumen dieses Simplex ist $\frac{1}{n!} \cdot \left| \det \begin{pmatrix} 1 & \cdots & 1 \\ x_0 & \cdots & x_n \end{pmatrix} \right|$. Mit der Cramerschen Regel, der Hadamard Ungleichung und viel Rechnen lässt sich dieses (sehr konservativ) durch $\frac{1}{(nL)^{3n^2}}$ von unten abschätzen.

Satz 6.8. *Sei $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ein Polytop mit $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, und $\dim P = n$. Sei zudem $L = \max\{|a_{ij}|, |b_i|\}$, $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$. Setze $R = (nL)^n$ und $N = (2n+1) \cdot 4n^2 \cdot ln(nL)$. Dann entscheidet die Ellipsoidmethode das Zulässigkeitsproblem für P korrekt in schwach-polynomieller Zeit.*

Beweis:
Wir zeigen, dass nach N Iterationen ein Volumen erreicht ist, so dass P leer sein muss, wenn zuvor kein zulässiger Punkt gefunden wurde.

Nach Voraussetzung liegt P in einer Kugel um 0 mit Radius $R = (nL)^n$. Diese wiederum liegt innerhalb eines „Quadranten“ mit Seitenlängen $2 \cdot (nL)^n$. Wir nutzen dies aus, um eine einfache obere Schranke von

$$\text{vol}(E_0) \leq 2(nL)^{n^2}$$

zu erhalten.
Für das Verhältnis der Volumina von E_0 und P gilt also:

$$\text{vol}\, E_0 / \text{vol}\, P \leq 2(nL)^{n^2} / \frac{1}{(nL)^{3n^2}} = 2(nL)^{4n^2}$$

Nach Satz 5.7 gilt nach k Iterationen:

$$\text{vol}\, E_k < \left(e^{-\frac{1}{2(n+1)}}\right)^k \text{vol}(E_0) = e^{-\frac{k}{2(n+1)}} \text{vol}(E_0)$$

Mit $N = (2n+1) \cdot 4n^2 \cdot \ln(nL)$ gilt also:

$$\text{vol}\, E_N < e^{-\frac{N}{2(n+1)}} \text{vol}(E_0) = e^{-\frac{(2n+1) \cdot 4n^2 \cdot \ln(nL)}{2(n+1)}} \text{vol}(E_0) = \frac{1}{(nL)^{4n^2}} \text{vol}(E_0)$$

Die Anzahl der Iterationen ist also *schwach*-polynomiell bzgl. der Eingabelänge des Systems $Ax \leq b$, d.h. n, m, und $\log_2 L$.

BEMERKUNG 6.9. *Der hier vorgestellte Algorithmus ist an einer Stelle sehr idealisiert. Wir haben angenommen, dass wir mit unendlicher Genauigkeit rechnen können. In der Definition von b, das zur Bestimmung von E_{k+1} verwendet wird, wird jedoch eine Wurzel gezogen. Der Algorithmus kann so angepasst werden, dass er mit vorgegebener Bitkomplexität läuft. Die Analyse wird dann aber deutlich technischer.*

BEMERKUNG 6.10. *Es kann passieren, dass es exponentiell viele Ungleichungen bezogen auf die Anzahl der Variablen gibt (m ist von der Ordnung 2^n). Die wahre Tragweite der vorherigen Beweise wird erst dadurch deutlicher, dass es dann jedoch immer noch möglich sein kein, das Trennungsproblem in polynomieller Zeit (bzgl. n) zu lösen. D.h. es ist möglich, Probleme in polynomieller Zeit zu lösen, deren entsprechende Beschreibung man nicht einmal in polynomieller Zeit als Eingabe an den Algorithmus übergeben kann!*

BEISPIEL 6.11. *Sei $G = (V, E)$ ein ungerichteter Graph, und $x \in \mathbb{R}^E$ mit $0 \leq x_e \leq 1$ für alle $e \in E$. Betrachte die folgenden „Zusammenhangsbedingungen“:*

$$\sum_{e \in \delta(S)} x_e \geq 1 \quad \text{für alle } \emptyset \neq S \subsetneq V$$

Die Anzahl dieser Ungleichungen ist exponentiell bzgl. $|V|$. Ob (mindestens) eine *davon durch ein gegebenes $x^* \in \mathbb{R}^E$ verletzt wird, kann jedoch in polynomieller Zeit entschieden werden (etwa durch Berechnung höchstens $|V|$ Minimum-s-t-Schnitte).*

Man kann sogar zeigen:

SATZ 6.12. *(Grötschel, Lovász, Schrijver, 1981; basierend auf Ellipsoidmethode) Das Optimierungsproblem lässt sich in (schwach-)polynomieller Zeit lösen genau dann, wenn sich das Trennungsproblem in (schwach-)polynomieller Zeit lösen lässt.*

Für den Beweis verweisen wir auf den Originalartikel der Autoren. Tatsächlich kann man die Ellipsoidmethode so erweitern, dass im Fall $x_k \in K$ der Zielfunktionsvektor $c \in \mathbb{R}^n$ als Schnitthyperebene benutzt wird, und auch für dieses Verfahren eine polynomielle Laufzeit bis zur ε-Approximation des Optimalwerts nachweisen. Die Rückrichtung, also dass man mit einem polynomiellen Algorithmus zur Lösung des Optimierungsproblems auch in polynomieller Zeit separieren kann, kann man z.B. mit Hilfe polarer Mengen beweisen.
Abschließend verweisen wir auf Innere-Punkte-Verfahren als die *praktischen* und zunehmend verwendeten und an Bedeutung gewinnenden (schwach-)polynomiellen Verfahren zur Lösung von LPs und anderen konvexen Optimierungsproblemen.

Teil C

Einführung in die ganzzahlige Optimierung

KAPITEL VII

Ganzzahlige lineare Optimierung & vollständige Unimodularität

1. Ganzzahlige lineare Programme

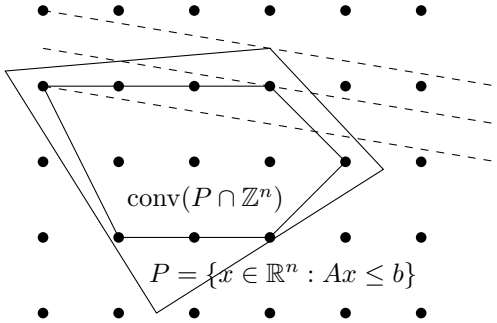
Viele Optimierungsprobleme des Operations Research lassen sich als *ganzzahlige lineare Programme* formulieren.

DEFINITION 1.1. *Ein* ganzzahliges lineares Programm *(in Standardform) ist von der Form:*

$$\begin{aligned} \max\ & c^\mathsf{T} x \\ & Ax \leq b, \\ & x \in \mathbb{Z}^n \quad (\text{„}x \text{ ganzzahlig}\text{“}) \end{aligned}$$

wobei $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ und $b \in \mathbb{R}^m$ gegeben sind. Ein ganzzahliges lineares Programm nennen wir auch ILP (= „integer linear program“) oder nur IP.

Geometrisch bedeutet dies, dass wir eine lineare Funktion über die Menge $P \cap \mathbb{Z}^n$, wobei $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ein Polyeder ist, maximieren.



Im Allgemeinen gilt: ILP kann *wahrscheinlich nicht effizient* (d.h. in polynomieller Zeit) gelöst werden (Äquivalent zum P $\neq$ NP-Problem).
Klar ist aber auch: Wir können ILP mit den uns bereits bekannten Methoden optimal lösen, wenn wir wissen, dass jede Ecke des dem Optimierungsproblem zugrundeliegenden Polyeders einer ganzzahligen Lösung entspricht.
In diesem Kapitel wollen wir uns mit den „glücklichen“ Fällen beschäftigen, bei denen dies der Fall ist.

2. Vollständig unimodulare Matrizen

DEFINITION 2.1. *Eine Matrix $A \in \mathbb{R}^{m\times n}$ heißt* vollständig unimodular *(VU), falls jeder ihrer Minoren (Determinanten quadratischer Teilmatrizen) gleich $0, -1$ oder $+1$ ist. Insbesondere ist auch jeder Eintrag $A_{ij} \in \{0, -1, +1\}$.*

BEMERKUNG 2.2. *Falls $A \in \mathbb{R}^{m\times n}$ vollständig unimodular ist, so sind auch die Matrizen A^T, $\begin{bmatrix} A & I \end{bmatrix}$ und $\begin{bmatrix} A \\ I \end{bmatrix}$ vollständig unimodular.*

SATZ 2.3. *Sei $A \in \mathbb{Z}^{m\times n}$ vollständig unimodular und sei $b \in \mathbb{Z}^m$. Dann ist jede Ecke z des Polyeders $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ganzzahlig, d.h. es gilt $z \in \mathbb{Z}^n$.*

DEFINITION 2.4. *Ein Polyeder $P \subseteq \mathbb{R}^n$ heißt* ganzzahlig*, falls für alle $c \in \mathbb{R}^n$, für die $\sup\{c^\mathsf{T}x : x \in P\}$ endlich ist, das Maximum an einem ganzzahligen Vektor angenommen wird.*

Falls P ganzzahlig ist, dann gilt also

$$\max\{c^\mathsf{T}x : x \in P,\ x \in \mathbb{Z}^n\} = \max\{c^\mathsf{T}x : x \in P,\ x \in \mathbb{R}^n\}$$

falls das Maximum angenommen wird, d.h. die Ganzzahligkeitsbedingung kann weggelassen werden.

SATZ 2.5. *Sei $A \in \mathbb{R}^{m\times n}$ eine vollständig unimodulare Matrix und sei $b \in \mathbb{Z}^m$. Dann ist $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ein ganzzahliges Polyeder.*

KOROLLAR 2.6. *Sei $A \in \mathbb{R}^{m\times n}$ eine vollständig unimodulare Matrix und sei $b \in \mathbb{Z}^m$ und $c \in \mathbb{Z}^n$. Dann haben die beiden linearen Programme*

$$\max\{c^\mathsf{T}x : Ax \leq b\} = \min\{b^\mathsf{T}y : y \geq 0,\ A^\mathsf{T}y = c\}$$

ganzzahlige Lösungen, falls die Optima endlich sind.

3. Vollständig unimodulare Matrizen und bipartite Graphen

DEFINITION 3.1. *Sei $G = (V, E)$ ein ungerichteter Graph. Die* Inzidenzmatrix *$A \in \mathbb{R}^{V\times E}$ von G ist definiert durch*

$$A_{v,e} = \begin{cases} 1, & \text{falls } v \in e, \\ 0, & \text{sonst.} \end{cases}$$

für alle $v \in V$ und alle $e \in E$.
D.h. jede Spalte von A enthält exakt zwei 1-Einträge.

SATZ 3.2. *Sei $G = (V, E)$ ein ungerichteter Graph. Dann ist G bipartit genau dann, wenn seine Inzidenzmatrix $A \in \mathbb{R}^{V\times E}$ vollständig unimodular ist.*

KOROLLAR 3.3. *(Matching-Theorem von König, vgl. Satz III.3.2)*
Sei $G = (V, E)$ ein bipartiter Graph. Dann gilt

$$\begin{aligned} \nu(G) &= \max\{|M| : M \subseteq E \text{ Matching in } G\} \\ &= \min\{|U| : U \subseteq V \text{ Knotenüberdeckung in } G\} \\ &= \tau(G). \end{aligned}$$

KOROLLAR 3.4. *(Theorem von Egerváry, 1931)*
Sei $G = (V, E)$ ein bipartiter Graph und $w \in \mathbb{Z}^E$ eine ganzzahlige Gewichtsfunktion. Dann ist

$$\begin{aligned} \nu_w(G) &= \max\{w(M) : M \subseteq E \text{ Matching in } G\} \\ &= \min\left\{\sum_{v\in V} y_v : y \in \mathbb{Z}^V,\ y \geq 0,\ y_u + y_v \geq w(\{u, v\})\ \forall\{u, v\} \in E\right\}. \end{aligned}$$

DEFINITION 3.5. *Sei $G = (V, E)$ ein ungerichteter Graph. Das Matchingpolytop von G ist die konvexe Hülle der charakteristischen Vektoren von Matchings in G:*

$$M(G) = \operatorname{conv}\{\chi^M : M \subseteq E \text{ Matching in } G\} \subseteq \mathbb{R}^E,$$

wobei der charakteristische Vektor χ^M eines Matchings $M \subseteq E$ in G definiert ist als

$$(\chi^M)_e = \begin{cases} 1, & \text{falls } e \in M, \\ 0, & \text{sonst.} \end{cases}$$

KOROLLAR 3.6. *Sei $G = (V, E)$ ein bipartiter Graph. Dann gilt*

$$M(G) = \{x \in \mathbb{R}^E : x \geq 0,\ Ax \leq 1\},$$

wobei A die Inzidenzmatrix von G ist.

BEMERKUNG 3.7. *Im allgemeinen, nicht-bipartiten, Fall gilt nur die Inklusion*

$$M(G) \subseteq \{x \in \mathbb{R}^E : x \geq 0,\ Ax \leq 1\}$$

und somit auch nur

$$\begin{aligned}\nu(G) &\leq \max\left\{1^\mathsf{T} x : x \in \mathbb{R}^E,\ x \geq 0,\ Ax \leq 1\right\} \\ &= \min\left\{1^\mathsf{T} y : y \in \mathbb{R}^V,\ y \geq 0,\ y^\mathsf{T} A \geq 1\right\} \\ &\leq \tau(G).\end{aligned}$$

Analog im Fall gewichteter Matchings.

BEISPIEL 3.8. *Für $G =$*  *ist*

$$M(G) = \operatorname{conv}\left\{ \begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix} \right\},\ \textit{jedoch}$$

$$\begin{pmatrix}\frac{1}{2}\\\frac{1}{2}\\\frac{1}{2}\end{pmatrix} \in \{x \in \mathbb{R}^E : x \geq 0, Ax \leq 1\},\ \textit{aber}\ \begin{pmatrix}\frac{1}{2}\\\frac{1}{2}\\\frac{1}{2}\end{pmatrix} \notin M(G).$$

Also ist $\nu(G) = 1 \leq 1.5 = \max\{1^T x : x \in \mathbb{R}^E, x \geq 0, Ax \leq 1\} \leq 2 = \tau(G)$.

4. Vollständig unimodulare Matrizen und gerichtete Graphen

DEFINITION 4.1. *Die* Inzidenzmatrix *eines gerichteten Graphen $D = (V, A)$ ist die Matrix $M \in \mathbb{R}^{V \times A}$, die durch*

$$M_{v,a} = \begin{cases} +1, & \textit{falls } a \in \delta^{in}(v), \quad \xrightarrow{\ a\ } v \\ -1, & \textit{falls } a \in \delta^{out}(v), \quad v \xrightarrow{\ a\ } \\ 0, & \textit{sonst.} \end{cases}$$

definiert ist.

Jede Spalte von M enthält genau eine $+1$ und genau eine -1.

SATZ 4.2. *Die Inzidenzmatrix eines gerichteten Graphen ist vollständig unimodular.*

KOROLLAR 4.3. *(Max-Flow-Min-Cut Theorem, vgl. Satz II.2.1)*
Seien $D = (V, A)$ ein gerichteter Graph, s, $t \in V$, und $c : A \to \mathbb{R}_{\geq 0}$ eine Kapazitätsfunktion. Dann gilt

$$\max\{\text{value}(f) : f \in \mathbb{R}^A_{\geq 0}\ s\text{-}t\text{-}\textit{Fluss}, f \leq c\} = \min\{c(\delta^{out}(U)) : U \subseteq V, s \in U, t \notin U\}.$$

5. Charakterisierung von vollständig unimodularen Matrizen

SATZ 5.1. *(Hoffman, Kruskal, 1956) Es sei $A \in \mathbb{Z}^{m \times n}$ eine ganzzahlige Matrix. Dann ist A vollständig unimodular genau dann, wenn für alle $b \in \mathbb{Z}^m$ das Polyeder $P = \{x \in \mathbb{R}^n : x \geq 0, Ax \leq b\}$ ganzzahlig ist.*

SATZ 5.2. *Die Problemstellung zu entscheiden, ob eine gegebene Matrix $A \in \mathbb{R}^{m \times n}$ vollständig unimodular ist, kann in polynomieller Zeit gelöst werden.*

KAPITEL VIII

Branch-and-Cut für Ganzzahlige Lineare Programme

Wie bereits im vorherigen Kapitel, studieren wir auch im Folgenden *ganzzahlige* lineare Programme (ILPs) der Form $\max\{c^\mathsf{T}x : Ax \leq b, x \in \mathbb{Z}^n\}$. Wir betrachten nun jedoch den allgemeinen Fall, dass das durch $A \in \mathbb{R}^{m\times n}$ und $b \in \mathbb{R}^m$ definierte Polyeder $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ nicht notwendigerweise ganzzahlig ist.
Immerhin: Geometrisch ist klar, dass P die gesuchten ganzzahligen Lösungen zumindest noch immer *enthält* (vgl. auch das Beispiel des 2D-Polytops zu Beginn von Kapitel VII). Dies legt zunächst einmal die folgende Definition nahe.

DEFINITION 0.1. *Die* ganzzahlige Hülle *eines Polyeders* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$, *mit* $A \in \mathbb{R}^{m\times n}$ *und* $b \in \mathbb{R}^m$, *ist*

$$P_I = \text{conv}(P \cap \mathbb{Z}^n) = \text{conv}\{x \in \mathbb{Z}^n : Ax \leq b\}.$$

Die Inklusionsbeziehung $P_I \subseteq P$ ist eng verbunden mit dem Begriff der Relaxierung.

DEFINITION 0.2. *Für ein ILP* $\sup\{c^\mathsf{T}x : Ax \leq b, x \in \mathbb{Z}^n\}$ *heißt das lineare Programm* $\sup\{c^\mathsf{T}x : Ax \leq b, x \in \mathbb{R}^n\}$ *die zu dem ILP gehörende* LP-Relaxierung. *Sie entsteht aus diesem durch Vernachlässigung aller Ganzzahligkeitsrestriktionen.*

BEOBACHTUNG 0.3.

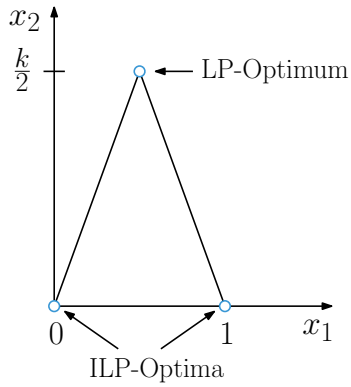
(1) *Jede zulässige Lösung von (ILP) ist auch zulässig für die zugehörige LP-Relaxierung.*
(2) *Der Wert einer Optimallösung der LP-Relaxierung, also bei Optimierung über P, liefert (daher) eine* obere Schranke *für den Wert einer Optimallösung von (ILP), also bei Optimierung über P_I, d.h. es gilt*

$$\sup\{c^\mathsf{T}x : Ax \leq b, x \in \mathbb{Z}^n\} \leq \sup\{c^\mathsf{T}x : Ax \leq b, x \in \mathbb{R}^n\}.$$

Problem: Die Abweichung der beiden Optimalwerte kann beliebig groß sein, wie nachfolgendes Beispiel zeigt:

BEISPIEL 0.4.

$$\begin{array}{lrcll} \max & & x_2 & & \\ & -kx_1 & + x_2 & \leq & 0 \\ & kx_1 & + x_2 & \leq & k \\ & & x_1, x_2 & \geq & 0 \\ & & x_1, x_2 & & \textit{ganzzahlig} \end{array}$$



Wert von (LP): $\frac{k}{2}$ *für* $k > 0$ *beliebig*
Wert von (ILP): 0

Außerdem zeigt dieses Beispiel die (komplexitätstheoretisch naheliegende) Tatsache, dass uns auch die oft erste Idee des Rundens von Lösungen i.A. nicht weiterhilft: Für $k > 1$ *sowohl ungerade als auch gerade, sind alle möglichen Rundungen von* $\begin{pmatrix} 1/2 \\ k/2 \end{pmatrix}$, *also die Vektoren* $\begin{pmatrix} 0 \\ \lfloor k/2 \rfloor \end{pmatrix}$ *und* $\begin{pmatrix} 0 \\ \lceil k/2 \rceil \end{pmatrix}$, *sowie* $\begin{pmatrix} 1 \\ \lfloor k/2 \rfloor \end{pmatrix}$ *und* $\begin{pmatrix} 1 \\ \lceil k/2 \rceil \end{pmatrix}$ *unzulässig für ILP.*

Trotz dieser Schwierigkeiten können wir uns mit geeigneten Zusatztechniken ausgehend von einer Optimallösung der LP-Relaxierung der des ILPs sukzessive annähern. Zwei elementare konzeptionelle Ansätze, dies im Falle einer also nicht ganzzahligen Lösung x^* der LP-Relaxierung von (ILP) zu tun sind das Thema dieses Kapitels.

1. Etwas mehr (rationale) Polyedertheorie

Definition 1.1. *Sei* $P \subseteq \mathbb{R}^n$ *ein Polyeder und* H *eine Stützhyperebene von* P, *also* $P \subseteq H^-$, $P \cap H \neq \emptyset$. *Dann heißt* $F = P \cap H$ Seitenfläche *von* P.

Lemma 1.2. *Sei* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *ein Polyeder. Dann gilt:* $F \subseteq P$ *ist eine Seitenfläche von* P *genau dann, wenn* $F \neq \emptyset$ *und* $F = \{x \in P : A'x = b'\}$ *für ein Teilsystem* $A'x \leq b'$ *von* $Ax \leq b$. *Außerdem enthält jede Seitenfläche von* P *einen ganzzahligen Vektor genau dann, wenn* P *ganzzahlig ist.*

Lemma 1.3. *Sei* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *ein Polyeder und* F *eine inklusionsminimale Seitenfläche von* P. *Dann existiert ein Teilsystem* $A'x \leq b'$ *von* $Ax \leq b$, *so dass* $F = \{x \in \mathbb{R}^n : A'x = b'\}$.

Definition 1.4. *Ein Polyeder* $P \subseteq \mathbb{R}^n$ *heißt* rationales *Polyeder, falls* $A \in \mathbb{Q}^{m \times n}$ *und* $b \in \mathbb{Q}^m$, *mit* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$.

Bemerkung 1.5. *Falls* P *rational ist, so können wir auch zugehörige* $A' \in \mathbb{Z}^{m \times n}$ *und* $b' \in \mathbb{Z}^m$ *finden, so dass* $P = \{x \in \mathbb{R}^n : A'x \leq b'\}$.

Lemma 1.6. *Sei* $A \in \mathbb{Q}^{m \times n}$ *und* $b \in \mathbb{Q}^m$. *Dann gilt: Es existiert ein* $x \in \mathbb{Z}^n$ *mit* $Ax = b$ *genau dann, wenn für alle* $y \in \mathbb{Q}^m$: $y^\mathsf{T} A \in \mathbb{Z}^n \Rightarrow y^\mathsf{T} b \in \mathbb{Z}$.

Satz 1.7. *Sei* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *ein rationales Polyeder. Dann ist* P_I *ebenfalls ein (rationales) Polyeder.*

Satz 1.8. *Sei* $P \subseteq \mathbb{R}^n$ *ein rationales Polyeder. Dann gilt:* P *ist ganzzahlig genau dann, wenn jede rationale Stützhyperebene von* P *einen ganzzahligen Vektor enthält.*

Korollar 1.9. *Sei* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *ein rationales Polyeder. Dann ist* P_I *ganzzahlig, und es gilt* $\max\{c^\mathsf{T} x : x \in P_I\} = \max\{c^\mathsf{T} x : x \in P,\ x \in \mathbb{Z}^n\}$, *falls* $\max < \infty$.

Bemerkung 1.10. *Aus Satz 1.7 folgt unmittelbar, dass für die ganzzahlige Hülle* P_I *eines rationales Polyeders* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *auch wieder eine Darstellung der Form*

$$P_I = \{x \in \mathbb{R}^n : \tilde{A}x \leq \tilde{b}\},$$

mit $\tilde{A} \in \mathbb{Z}^{\tilde{m} \times n}$ *und* $\tilde{b} \in \mathbb{Z}^{\tilde{m}}$ *existiert. Kannten wir diese, könnten wir Optimierungsprobleme über* P_I *also wieder als LP lösen.*
Im Allgemeinen wird sich eine solche Beschreibung von P_I *jedoch unserer Kenntnis entziehen und/oder die Anzahl der dafür benötigten Ungleichungen* $\tilde{m}$ *zu groß (von exponentieller Ordnung) sein, um diese (direkt) in einem LP zu berücksichtigen.*

Aber: Es werden auch längst nicht alle Halbräume, die P_I definieren, zur Lösung eines ILPs über P bzw. P_I benötigt. Wir suchen nun also „billige" (d.h. algorithmisch leicht zu ermittelnde) rationale Halbräume H mit $P_I \subseteq H^-$ und $P \not\subseteq H^-$. Das iterative, systematische Hinzufügen dieser ist die grundlegende Idee von sog. Schnittebenenverfahren (engl. 'cutting plane methods').

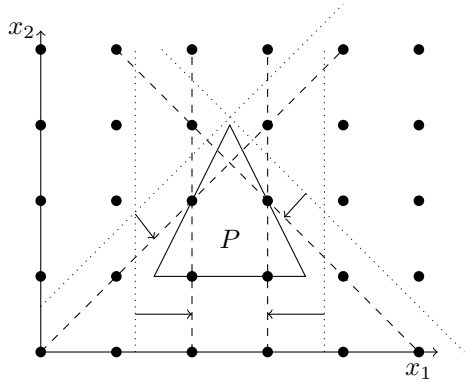
2. Der Chvátal-Gomory Abschluss

DEFINITION 2.1. *Sei P ein rationales Polyeder. Der Chvátal-Gomory (CG) Abschluss von P ist*

$$P' = \bigcap_{\substack{H \text{ rationaler Halbraum,} \\ H \supseteq P}} H_I,$$

wobei $H_I = \operatorname{conv}(H \cap \mathbb{Z}^n)$.

BEMERKUNG 2.2. *Für $H = \{x \in \mathbb{R}^n : a^\mathsf{T} x \leq b\}$ mit $a \in \mathbb{Q}^n \setminus \{0\}, b \in \mathbb{Q}$ kann H_I leicht berechnet werden: Wir können o.B.d.A. annehmen, dass $a \in \mathbb{Z}^n \setminus \{0\}$. Dann erhalten wir H_I, indem wir H in Richtung $-a$ verschieben, bis ganzzahlige Punkte auf dem Rand liegen. Diese Punkte erfüllen also $a^\mathsf{T} x = \lfloor b \rfloor$. Also ist $H_I = \{x \in \mathbb{R}^n : a^\mathsf{T} x \leq \lfloor b \rfloor\}$. Da $P \subseteq H$ impliziert, dass $P_I \subseteq H_I$, gilt $P_I \subseteq P' \subseteq P$.*



Frage: Wieso kann $x_2 \leq 2$ nicht erzeugt werden?

SATZ 2.3. *Sei $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ein rationales Polyeder. Dann gilt*

$$P' = P \cap R,$$

wobei $R = \{x \in \mathbb{R}^n : \mu^\mathsf{T} Ax \leq \lfloor \mu^\mathsf{T} b \rfloor,\ \mu^\mathsf{T} A \in \mathbb{Z}^n,\ 0 \leq \mu < 1\}$.

DEFINITION 2.4. *Seien $A \in \mathbb{Q}^{m \times n}$ und $b \in \mathbb{Q}^m$. Eine Ungleichung $\mu^\mathsf{T} Ax \leq \lfloor \mu^\mathsf{T} b \rfloor$, mit $\mu \in \mathbb{R}^m$, $\mu > 0$, so dass $\mu^\mathsf{T} A \in \mathbb{Z}^n$, heißt* Chvátal-Gomory (CG) Schnitt.

KOROLLAR 2.5. *Die Anzahl der benötigten CG Schnitte, um den CG Abschluss eines rationalen Polyeders $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ zu bilden, ist endlich.*

KOROLLAR 2.6. *Sei $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ein rationales Polyeder. Dann ist auch der CG Abschluss P' von P ein rationales Polyeder.*

Dies bedeutet, dass wir das Verfahren iterieren, d.h. den CG-Abschluss P'' von P' berechnen können, usw., so dass $P \supseteq P' \supseteq P'' \supseteq \cdots \supseteq P_I$. Folgender Satz zeigt, dass die Teilmengenbeziehungen ab einem gewissen Punkt nicht mehr strikt sind.

SATZ 2.7. *Sei P ein rationales Polyeder, $P^{(0)} := P$ und $P^{(i+1)} := (P^{(i)})'$. Dann existiert ein $t \in \mathbb{N}$, so dass $P^{(t)} = P_I$. Das kleinste t mit $P^{(t)} = P_I$ heißt der* Chvátal-Rang *von P.*

Wir wollen nun zeigen, dass, gegeben eine optimale Lösung der LP-Relaxierung, das Trennungs- bzw. Separationsproblem für CG-Schnitte effizient gelöst werden kann. Dazu nehmen wir Nicht-Negativitätsbedingungen für x an. Dies ist keine Einschränkung, da für $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, und $c \in \mathbb{R}^n$ gilt:

$$\sup\{c^\mathsf{T}x : Ax \le b\} = \sup\{c^\mathsf{T}(x-y) : x, y \ge 0, A(x-y) \le b\}$$

Unter Nicht-Negativitätsbedingungen muss μ zudem nicht notwendigerweise direkt so gewählt werden, dass $\mu^T A \in \mathbb{Z}^n$, sondern kann ebenfalls gerundet werden.

LEMMA 2.8. *Seien $P = \{x \in \mathbb{R}^n : x \ge 0, Ax \le b\}$, mit $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m$, und $\mu \ge 0$. Dann gilt:*

$$\lfloor \mu^\mathsf{T} A \rfloor x \le \lfloor \mu^\mathsf{T} b \rfloor \text{ für alle } x \in P_I.$$

LEMMA 2.9. *Sei $P = \{x \in \mathbb{R}^n : x \ge 0,\ Ax \le b\}$ mit $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m$, und $\mu \in \mathbb{R}^m$. Dann gilt:*

$$\lfloor \mu^\mathsf{T} A \rfloor x - \lfloor \mu^\mathsf{T} \rfloor Ax \le \lfloor \mu^\mathsf{T} b \rfloor - \lfloor \mu^\mathsf{T} \rfloor b \text{ für alle } x \in P_I.$$

Mit Hilfe von Lemma 2.9 und der darin beschriebenen Technik, können wir also auch für $\mu \not\ge 0$ eine gültige Ungleichung ableiten.

DEFINITION 2.10. *(Separationsproblem (bzw. Trennungsproblem) für CG-Schnitte)*
Eingabe: Eine Ecke $x^* \in \mathbb{R}^n$ von $P = \{x \in \mathbb{R}^n : Ax \le b,\ x \ge 0\}$, $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m$.
Ausgabe: „$x^* \in \mathbb{Z}^n$“, oder ein CG-Schnitt der x^* von P' trennt, d.h. eine Ungleichung $\lfloor \mu^\mathsf{T} A \rfloor x - \lfloor \mu^\mathsf{T} \rfloor Ax \le \lfloor \mu^\mathsf{T} b \rfloor - \lfloor \mu^\mathsf{T} \rfloor b$ mit $\lfloor \mu^\mathsf{T} A \rfloor x^* - \lfloor \mu^\mathsf{T} \rfloor Ax^* > \lfloor \mu^\mathsf{T} b \rfloor - \lfloor \mu^\mathsf{T} \rfloor b$.

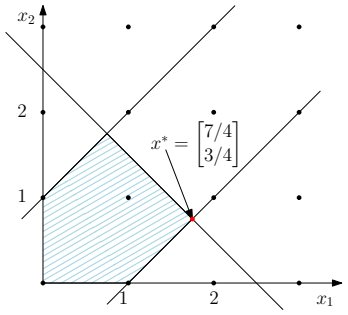
SATZ 2.11. *Sei $P = \{x \in \mathbb{R}^n : Ax \le b,\ x \ge 0\}$ ein Polyeder mit $A \in \mathbb{Z}^{m\times n}$, und $b \in \mathbb{Z}^m$. Sei zudem mit x^* eine Ecke von P gegeben. Dann kann das Separationsproblem für CG-Schnitte bzgl. P' und x^* in polynomieller Zeit (bzgl. m und n) gelöst werden.*

BEISPIEL 2.12.

Seien $A = \begin{bmatrix} 2 & 2 \\ 1 & -1 \\ -1 & 1 \end{bmatrix}$, $b = \begin{pmatrix} 5 \\ 1 \\ 1 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$,

und $P = \{x \in \mathbb{R}^2 : Ax \le b,\ x \ge 0\}$.

Eine Optimallösung ist $\begin{bmatrix} x_1^* \\ x_2^* \end{bmatrix} = \begin{bmatrix} 7/4 \\ 3/4 \end{bmatrix}$.



Betrachte das System $Ax + Is = b$:

$$\begin{array}{rcrcrcrcl} 2x_1 & + & 2x_2 & + & s_1 & & & = & 5 \\ x_1 & - & x_2 & & & + & s_2 & = & 1 \\ -x_1 & + & x_2 & & & & + \ s_3 & = & 1 \end{array}$$

Einsetzen von x_1^* und x_2^* liefert $s_1 = 0$, $s_2 = 0$, und $s_3 = 2$. Es ist also $N = \{3, 4\}$ und somit $B = \{1, 2, 5\}$.
$x_1^* \notin \mathbb{Z}$, daher betrachten wir die 1. Gleichung. Eine diagonalisierte Matrix A'_B bedeutet, dass in dieser Zeile nur noch x_1 aus B stehen bleibt: Wir erreichen dies, indem wir die Zeile durch 4 teilen, sowie die 2. Gleichung $\frac{1}{2}$-fach addieren, d.h. $\mu_1 = \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & 0 \end{pmatrix}^\mathsf{T}$.

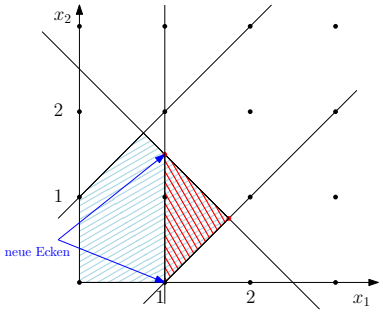
Wir erhalten:

$\lfloor \mu_1^\mathsf{T} A' \rfloor x' \leq \lfloor \mu_1^\mathsf{T} b \rfloor$

$$= \left\lfloor \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & 0 \end{pmatrix}^\mathsf{T} \begin{bmatrix} 2 & 2 & 1 & 0 & 0 \\ 1 & -1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 & 1 \end{bmatrix} \right\rfloor x' \leq \left\lfloor \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & 0 \end{pmatrix}^\mathsf{T} \begin{bmatrix} 5 \\ 1 \\ 1 \end{bmatrix} \right\rfloor$$

$$= \left\lfloor \begin{pmatrix} 1 & 0 & \frac{1}{4} & \frac{1}{2} & 0 \end{pmatrix}^\mathsf{T} \right\rfloor x' \leq \left\lfloor \frac{7}{4} \right\rfloor$$

$$= x_1 + \lfloor \frac{1}{4} \rfloor s_1 + \lfloor \frac{1}{2} \rfloor s_2 \leq \left\lfloor \frac{7}{4} \right\rfloor = x_1 \leq 1$$



(Ende Beispiel)

Das Besondere hier ist, dass also *immer* ein CG-Schnitt bestimmt werden kann, *solange die Lösung der aktuellen LP-Relaxierung nicht ganzzahlig ist* (oder Unzulässigkeit bzgl. ganzzahliger Lösungen festgestellt wurde), und dieser Schnitt immer einen Fortschritt in der Annäherung an P_I liefert. Daraus bzw. vielmehr aus Satz 2.7, folgt sofort, dass folgender Algorithmus (ILP) in endlicher Zeit löst:

ALGORITHMUS 2.13. *„Schnittebenenverfahren für ILP“*

Eingabe : $A \in \mathbb{Z}^{m \times n}, b \in \mathbb{Z}^m$, $c \in \mathbb{R}^n$
Ausgabe : $x^* \in \mathbb{Z}^n$ *mit* $c^\mathsf{T} x^* = \max\{c^\mathsf{T} x : Ax \leq b,\ x \in \mathbb{Z}^n\}$ *oder „*$Ax \leq b$ *hat keine ganzzahlige Lösung“.*
Setze $k = 0$, $A_0 = A$, *und* $b_0 = b$.
Wiederhole:
Setze $P_k = \{x \in \mathbb{R}^n : A_k x \leq b_k\}$.
Löse das LP $p^* = \max\{c^\mathsf{T} x : x \in P_k\}$.
Falls $p^* = -\infty$, *gib „*$Ax \leq b$ *hat keine ganzzahlige Lösung“ aus. STOP.*
Sei ansonsten x^* *die gefundene Optimallösung.*
Falls $x^* \in \mathbb{Z}^n$, *gib* x^* *aus, STOP.*
Andernfalls berechne einen CG-Schnitt $a^\mathsf{T} x \leq \beta$, *der* x^* *von* P'_k *trennt.*
Bilde A_{k+1} *aus* A_k *durch Hinzufügen der Zeile* a^T.
Bilde b_{k+1} *aus* b_k *durch Hinzufügen der Komponente* β.
Setze $k = k + 1$.

Anmerkung: Wir haben hier zur Einfachheit angenommen, dass das ILP nicht unbeschränkt ist. In diesem Fall wäre gleich das erste gelöste LP ebenfalls unbeschränkt.

BEMERKUNG 2.14. *Auch wenn das Separationsproblem für CG Schnitte* an Ecken *effizient gelöst werden kann, kann (vermutlich) weder eine Beschreibung von* P' *noch eine Lösung von ILP effizient bestimmt werden. Denn die komplexitätstheoretische Äquivalenz von Optimierung und Separierung (vgl. Satz VI.6.12) setzt polynomiell lösbare Trennungsprobleme für* beliebige *(von der Ellipsoidmethode gelieferte) Punkte voraus. Die in Algorithmus 2.13 auftretenden LPs wachsen zudem*

stetig an und können exponentiell groß werden. Zudem treten in der Praxis numerische Komplikationen auf, so dass das Verfahren nicht 1-zu-1 so angewendet wird. Dennoch ist die grundlegende und elegante Idee sich nur im Bereich des jeweiligen LP-Optimums an P' (bzw. P_I) anzunähern ein Schlüssel zur Praxistauglichkeit heutiger ILP-Lösungsverfahren, in denen angepasste Schnittebenenverfahren mit weiteren (wie z.B. der im folgenden Abschnitt thematisierten) Techniken kombiniert werden.

BEMERKUNG 2.15. *Von großer praktischer Bedeutung ist eine Unterklasse der CG-Schnitte, nämlich die sog. „$\{0, \frac{1}{2}\}$" (sprich 'Zero-Half') Cuts, bei denen $\mu^\mathsf{T} \in \{0, \frac{1}{2}\}$. Informell bedeutet dies, wir generieren eine Ungleichung aus dem Ursprungssystem, indem wir eine Ungleichung entweder gar nicht wählen, oder sie mit Faktor $\frac{1}{2}$ mit anderen Ungleichungen linearkombinieren (und dann die rechte Seite abrunden). Für viele bekannte kombinatorische Optimierungsprobleme sind Ungleichungen bekannt, die sich als Zero-Half-Cuts darstellen lassen, und deren zugehörige Hyperebenen inklusionsmaximale Seitenflächen der zugehörigen ganzzahligen Hülle bilden.*

BEMERKUNG 2.16. *Vor wenigen Jahren wurde gezeigt, dass der Chvátal-Gomory Abschluss von irrationalen Polyedern (bzw. sogar allgemeiner, von kompakten konvexen Mengen) ebenfalls ein rationales Polyeder ist. Einfacher noch kann man für jedes (irrationale) Polyeder P ein rationales Polyeder Q finden, so dass $P_I = Q_I$. Wähle dazu ein rationales Polyeder R das P enthält. Füge dann für jeden ganzzahligen Vektor $z \in R \setminus P$ einen rationalen Halbraum zur Beschreibung von R hinzu, so dass die zugehörige Hyperebene z von P trennt. Das resultierende Polyeder Q hat die gewünschten Eigenschaften.*

3. Branch-&-Bound und Branch-&-Cut

Wir haben nun bereits eine Methode kennengelernt, die uns Fortschritt bei der Lösung eines ILPs garantiert, d.h. insbesondere ausschließt, dass eine zuvor berechnete für das ILP unzulässige Lösung nicht noch einmal auftreten kann. Eine Weitere basiert auf der folgenden einfachen Grundidee:
Sei x^* eine Optimallösung von $\max\{c^\mathsf{T}x : Ax \leq b\}$ und $x^* \notin \mathbb{Z}^n$. Wähle eine Variable x_i, so dass $x_i^* \notin \mathbb{Z}$. Erstelle aus dem bisherigen zwei neue LPs, wovon eines die Ungleichung $x_i \leq \lfloor x_i^* \rfloor$, und das andere die Ungleichung $x_i \geq \lceil x_i^* \rceil$ beinhaltet. Eine ganzzahlige Optimallösung ist dann (sofern sie existiert) offensichtlich in mindestens einem der beiden neuen Teilprobleme (Polyeder) enthalten. Den Schritt der Verzweigung in Teilprobleme nennt man (engl.) „Branch". Ist die jeweilige Lösung erneut nicht ganzzahlig, wird das Verzweigen auf den beiden Teilproblemen nach demselben Prinzip fortgeführt. Man beachte, dass gilt:

$$\max\{c^\mathsf{T}x : Ax \leq b, x_i \leq \lfloor x_i^* \rfloor\} \leq \max\{c^\mathsf{T}x : Ax \leq b\} \text{ und auch}$$
$$\max\{c^\mathsf{T}x : Ax \leq b, x_i \leq \lceil x_i^* \rceil\} \leq \max\{c^\mathsf{T}x : Ax \leq b\}.$$

Der Zielfunktionswert des LPs vor dem „Branch" ist also eine obere Schranke für jedes der aus ihm erzeugten neuen LPs. Zudem liefert jedes LP wie gewohnt eine obere Schranke (engl. „Bound") für den Wert einer besten ganzzahligen Lösung:

$$\max\{c^\mathsf{T}x : Ax \leq b, x \in \mathbb{Z}^n\} \leq \max\{c^\mathsf{T}x : Ax \leq b\}$$

Wird daher im Verlauf des Verfahrens eine ganzzahlige Lösung x^* mit Zielfunktionswert $p^* = c^\mathsf{T}x^*$ gefunden, und haben alle noch offenen (noch nicht gelösten) LPs eine obere Schranke, die kleiner oder gleich p^* ist, so kann das Verfahren vorzeitig terminieren – die Optimalität von x^* ist bewiesen.
Wir skizzieren nun ein LP-basiertes Branch & Bound Verfahren für ILP angelehnt an die Darstellung in [Dakin, R.J., A tree search algorithm for mixed integer programming problems, Computer Journal 8 (1965) 250–255].

ALGORITHMUS 3.1. *„LP-basiertes Branch-&-Bound-Verfahren für ILP“*

Eingabe : $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$
Ausgabe : $x^* \in \mathbb{Z}^n$ *mit* $c^\mathsf{T}x^* = \max\{c^\mathsf{T}x : Ax \le b,\ x \in \mathbb{Z}^n\}$ *oder „*$Ax \le b$ *hat keine ganzzahlige Lösung“.*

(1) Initialisierung:

$P_0 := \max\{c^T x : Ax \le b\}$	*(erste LP-Relaxierung)*
$O_0 := \infty$	*(bekannte obere Schranke von* P_0*)*
$U := -\infty$	*(globale untere Schranke)*
$\tilde{x}$ *uninitialisiert*	*(beste ganzzahlige Lösung)*
$L := \{P_0\}$	*(Liste der aktiven Teilprobleme)*
$k := 0$	*(letzter vergebener Index)*

(2): Falls $L = \emptyset$ *STOP:*

$U = -\infty$	$\Rightarrow$	*Gib „*$Ax \le b$ *hat keine ganzzahlige Lösung“ aus.*
$U > -\infty$	$\Rightarrow$	*Gib* $\tilde{x}$ *als Optimallösung (mit Wert* U*) aus.*

(3): Wähle ein Teilproblem bzw. LP $P_\ell \in L$ *und setze* $L = L \setminus \{P_\ell\}$.
(4): Ist $O_\ell \le U$*, so gehe zu (2).*
(5): Löse das LP P_ℓ *(Bound).*
(6): Ist P_ℓ *unzulässig oder unbeschränkt, so gehe zu (2).*
Andernfalls sei x^* *eine Optimallösung.*
(7): Ist $c^\mathsf{T}x^* > U$ *und* $x^* \in \mathbb{Z}^n$*, setze* $U = c^\mathsf{T}x^*$, $\tilde{x} = x^*$.
(8): Ist $c^\mathsf{T}x^* > U$ *und* $\exists\ j \in \{1,\dots n\} : x_j^* \notin \mathbb{Z}$*, wähle ein solches und setze:*

$$P_{k+1} := P_\ell \cap \{x \in \mathbb{R}^n : x_j \le \lfloor x_j^* \rfloor\}, \quad O_{k+1} := c^\mathsf{T}x^*$$

$$P_{k+2} := P_\ell \cap \{x \in \mathbb{R}^n : x_j \ge \lceil x_j^* \rceil\}, \quad O_{k+2} := c^\mathsf{T}x^*$$

Setze $L = L \cup \{P_{k+1}, P_{k+2}\}$ *(Branch),* $k = k + 2$.
(9) Gehe zu (2).

BEMERKUNG 3.2. *Wird in Schritt (5) von Algorithmus 3.1 nicht nur* ein *LP gelöst, sondern ein (partielles) Schnittebenenverfahren verwendet, um die dort erhaltene obere Schranke (möglichst) zu verbessern, so spricht man von einem Branch-&-Cut-Verfahren. Beide Ansätze können zudem auf vielfältige Weise variiert, angepasst, und erweitert werden.*

BEISPIEL 3.3. *Anhand folgenden Beispiels (mit* n *ungerade gewählt) verdeutlicht man sich leicht, dass Algorithmus 3.1 eine Laufzeit exponentiell in der Anzahl der Variablen* n *haben kann.*

$$\begin{array}{rcl} \max \quad -x_{n+1} & & \\ 2x_1 + 2x_2 + \ldots + 2x_n + x_{n+1} & = & n \\ x_1, \ldots, x_{n+1} & \ge & 0 \\ x_1, \ldots, x_{n+1} & \le & 1 \\ x_1, \ldots, x_{n+1} & \in & \mathbb{Z} \end{array}$$

BEMERKUNG 3.4. *Eine praktische Umsetzung von Algorithmus 3.1 erfordert das Treffen diverser Entscheidungen, die flexibel bzw. problemangepasst möglich sein sollten, und großen Einfluss auf die (praktische) Laufzeit haben können, etwa:*

- *Die Wahl einer* Branching-Regel, *nach der die (fraktionale) Variable für das Branching in Schritt (8) ausgewählt wird.*
- *Die Wahl einer* Selection-Regel, *nach der in Schritt (3) das als nächste zu bearbeitende Teilproblem aus der Liste* L *ausgewählt wird.*

BEMERKUNG 3.5. *Die grundlegende Idee eines Branch & Bound Verfahrens kann wesentlich allgemeiner aufgefasst werden: Im Prinzip ist jede Wahl von Teilproblemen / Relaxierungen denkbar, die den Lösungsraum korrekt partitioniert.*

BEMERKUNG 3.6. *Auch die grundsätzliche Idee, Ungleichungen „nur bei Bedarf" hinzuzufügen, kann auf bereits bekannte (und ggf. für die ILP-Formulierung notwendige) Ungleichungsklassen übertragen werden. Dieser Ansatz ist insbesondere dann interessant, wenn bereits die Anzahl dieser Ungleichungen exponentiell in der Anzahl der Variablen – das zugehörige Separationsproblem aber in polynomieller Zeit lösbar ist (vgl. z.B. Beispiel VI.6.11). Man startet dann also nur mit einer Teilmenge der Restriktionen (z.B. sogar nur mit unteren und oberen Schranken $l \leq x \leq u$ für die Variablen), löst die Relaxierung, und anschließend das Separationsproblem für die bisher vernachlässigten Ungleichungen. Falls eine verletzte Ungleichung existiert, so wird diese wie beim skizzierten Schnittebenenverfahren hinzugefügt. Andernfalls ist die Relaxierung in Schritt (5) gelöst (ggf. ohne alle Ungleichungen explizit betrachtet zu haben). In der Praxis werden hinzugefügte Ungleichungen auch nach einigen Iterationen wieder aus dem LP entfernt, sofern sie nicht mit Gleichheit erfüllt sind.*