Lineare und Ganzzahlige Optimierung
(ADM II)

Skript

Rolf Möhring
WS 2010/11

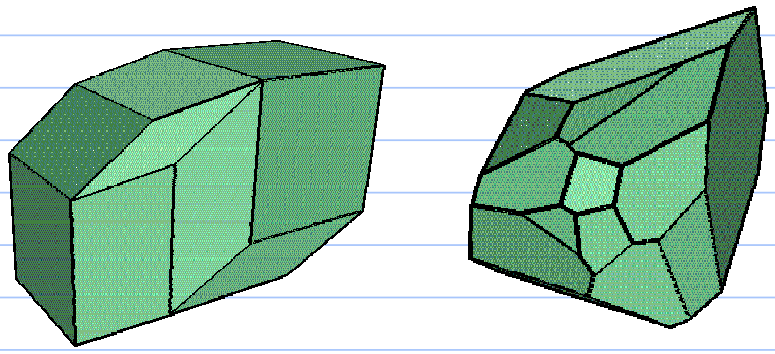
Contents 1-1

- **Zur Entstehung der ADM** ...
  - Junges Gebiet, hat Wurzeln in
    - Algebra, Graphentheorie, Kombinatorik
    - Informatik (Algorithmik)
    - Optimierung
  - Behandelt Optimierungsfragen bei Diskreten Strukturen
    - Graphen, Netzwerke
    - endliche Lösungsmenge
  - Anwendungen
    - Telekommunikations- und Verkehrsnetze
    - Logistik, Produktionsplanung, Standortoptimierung
    - ...

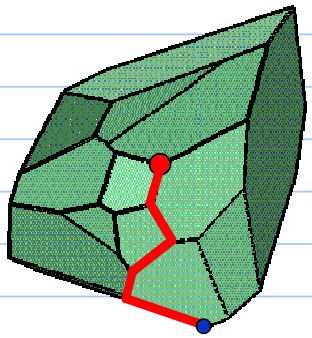
- Vorlesungszyklus ADM an der TU Berlin ...
  - Grundvorlesungen
    - Graphen- und Netzwerkalgorithmen (ADM I)

    - Lineare Optimierung (ADM II)
  - Vertiefung (ADM III) eine Vorlesung aus Katalog
    - Scheduling Probleme
    - Angewandte Netzwerkoptimierung
    - Polyedertheorie
    - ...
  - Seminar (teils bereits parallel zu ADM II oder ADM III)
  - Bachelorarbeit

- Lineare Optimierungsprobleme ...
  - lineare Zielfunktion, lineare Ungleichungen als Nebenbedingungen
    - Lineare Optimierung: $\min c^T x$ unter $Ax \leq b, x \geq 0$
  - Simplexalgorithmus
  - Dualität
  - Geometrie linearer Optimierungsprobleme
    - $Ax \leq b, x \geq 0$ definieren Polyeder

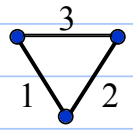


- Optimum wird auf Ecke angenommen
- Simplexalgorithmus durchläuft Ecken

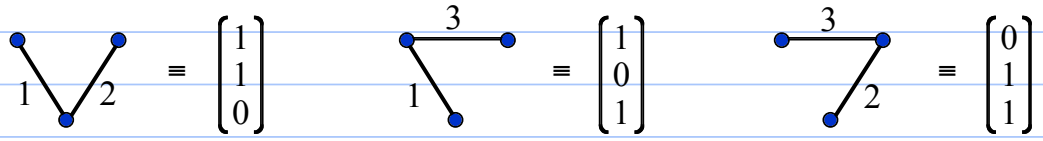


- Diskrete Probleme als Lineare Optimierungsprobleme ...
  - Polyedertheorie

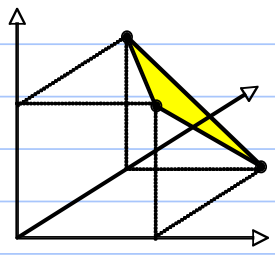
- diskrete Probleme als geometrische Probleme
  - minimal spannende Bäume als Vektoren
    - gegebener Graph G



- minimal spannende Bäume von G als Vektoren (Inzidenzvektoren)



- Konvexe Hülle der Inzidenzvektoren = Polytop (gelbe Menge)



Polytop = gelbe Menge

- Ermittlung minimal spannender Baum = lineare Optimierung über diesem Polytop

1.2 Inhalt von ADM II

- Ganzzahlige lineare Optimierung
  - Variablen dürfen nur ganzzahlige Werte annehmen
    - deutlich schwierigere Probleme
  - Lösungsverfahren
    - Lagrange Relaxation
    - Schnittebenenverfahren
    - LP-basierte Approximationsalgorithmen
    - ...
- Übungen mit Implementationsaufgaben

1.3 VL im WS 2010/11

- Torsten Gellert (Übung)
- Christoph Hansknecht (Tutorien)
- Webseite zur VL
  - @ http://www.math.tu-berlin.de/coga/teaching/wt08/adm2/
  - @ http://www.math.tu-berlin.de/coga/teaching/wt10/adm2/
  - @ Notebook: http://www.math.tu-berlin.de/~moehring/adm2/
- Literatur
  - C. H. Papadimitriou and K. Steiglitz

    Combinatorial Optimization: Algorithms and Complexity

    Prentice Hall, Englewood Cliffs, NJ, 1982

    Taschenbuch - 512 Seiten - Dover Publications

    Erscheinungsdatum: Juli 1998

    Auflage: Unabridged

    ISBN: 0486402584
  - B. Korte, J. Vygen:

    Combinatorial Optimization: Theory and Algorithms

Springer, 2000/2002/2006/2008

jetzt auch auf deutsch

- V. Chvátal

  Linear Programming

  Freeman, New York, 1983

- W. J. Cook, W. H. Cunningham, W. R. Pulleyblank und A. Schrijver

  Combinatorial Optimization

  Wiley 1998

- G. L. Nemhauser and L. A. Wolsey

  Integer and Combinatorial Optimization

  John Wiley & Sons, New York, 1988

- M. Grötschel, L. Lovász, and A. Schrijver

  Geometric Algorithms and Combinatorial Optimization

  Springer-Verlag, Berlin, 2nd ed., 1993

- D. S. Hochbaum, ed.

  Approximation Algorithms for NP-hard Problems

PWS Publishing Company, Boston, MA, 1997

- H. M. Salkin and K. Mathur

  Foundations of Integer Programming

  North-Holland, Amsterdam, 1989.

@ - R. J. Vanderbei

  Linear Programming: Foundations and Extensions

  Kluwer Acad. Publ., Dordrecht, 2nd ed., 2001.

  http://www.princeton.edu/~rvdb/LPbook/index.html

- als Enzyklopädie

  A. Schrijver:

  Combinatorial Optimization: Polyhedra and Efficiency

  Springer, 2003

  3 bändig mit 1881 Seiten, auch als CD erhältlich

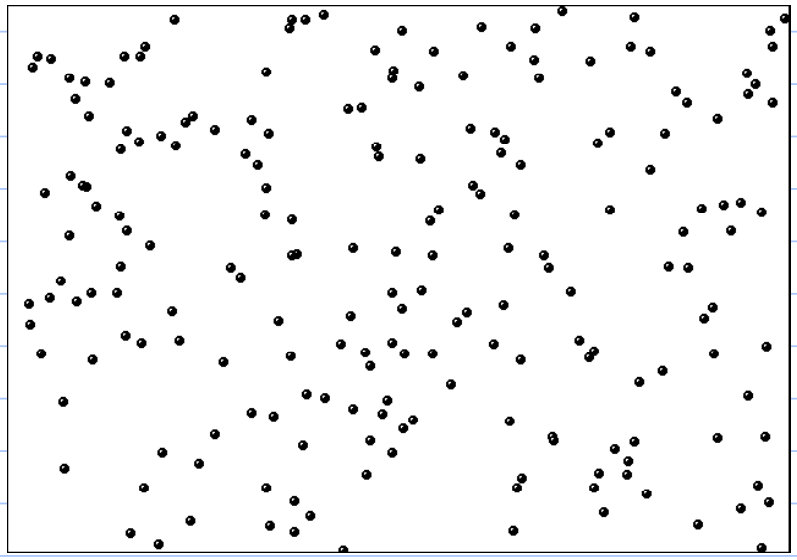
2.1 Beispiele

- Ein (NP-)Optimierungsproblem P_0 ist wie folgt definiert
  - Jede Instanz $I \in P_0$ hat einen Zulässigkeitsbereich S_I aus sogenannten Lösungen (solutions)
    - Zulässigkeit (Test $y \in S_I$) kann in polynomialer Zeit überprüft werden
  - Aufgabe:
    - bestimme für eine Instanz I und eine Zielfunktion $c : S_I \rightarrow Q$ (rationale Zahlen) eine Optimallösung $y = OPT(I)$

      d.h. $y \in S_I$ mit $OPT(I) = c(y) \leq c(x)$ für alle $x \in S_I$
      - $OPT(I)$ bezeichnet je nach Kontext die optimale Lösung oder den Zielfunktionswert der optimalen Lösung
    - eine solche Lösung heißt globales Optimum (globales Minimum) oder Optimum (Minimum)
  - Ein Algorithmus, der dies leistet, heißt exakt

- 2.1 Beispiel: Traveling Salesman Problem (TSP)
  - Instanz
    - vollständiger Graph K_n, $n \geq 3$
    - Kantenbewertung $c(e)$ mit rationalen Zahlen ≥ 0
  - Aufgabe

- Bestimme einen Hamilton Kreis C mit minimaler Länge
  - $c(C) = \sum_{e \in E(C)} c(e)$
- Beispiel einer Instanz
  - $G = K_n$ in der Ebene mit Euklidischem Abstand



  - Zwei zulässige Lösungen

- **Nachbarschaften**
  - Nachbarschaften sind für stetige Probleme in natürlicher Weise als ε-Umgebung bzgl. einer Norm definiert.

    Wie für diskrete Probleme?

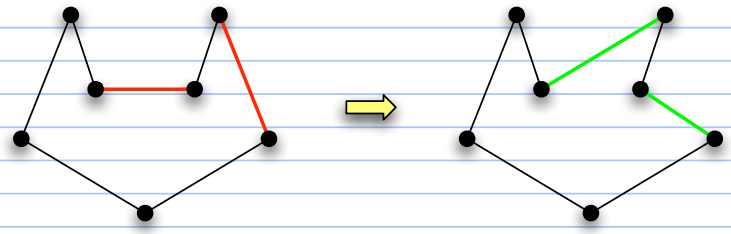
  - Eine Nachbarschaft für eine Problemklasse P_0 ist gegeben durch eine Abbildung

    $N_I : S_I \rightarrow 2^{S_I}$

    für jede Instanz $I \in P_0$

    $N_I(y)$ heißt die Nachbarschaft von $y \in S_I$. Ist I klar, so schreiben wir kurz $N(y)$
- 2.4 Beispiel: TSP
  - Definieren Nachbarschaft über 2-Tausch

    $N_2(y)$:= { $x \in S_I$ | x entsteht aus y durch Ersetzen von ≤ 2 Kanten aus y durch andere Kanten }

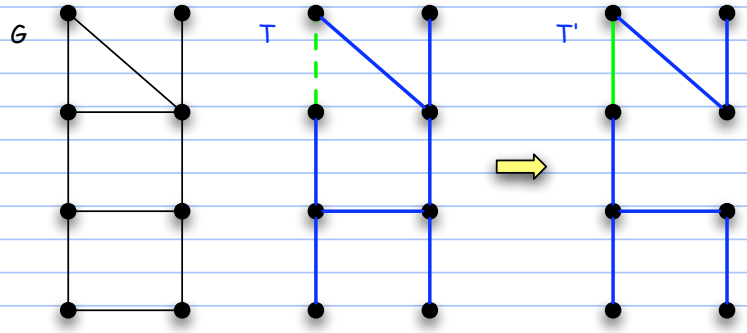


Verallgemeinerung auf $k \geq 2$ möglich, führt zu Nachbarschaft $N_k(y)$

2.5 Beispiel: MST

Nachbarschaft über Austausch einer Kante auf Fundamentalkreis definieren

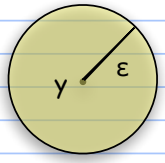
$N(y) := \{ x \in S_I \mid x$ entsteht aus y durch Hinzufügen einer Kante und Entfernen einer anderen auf dem entstehenden Kreis $\}$



2.6 Beispiel: LP

Nachbarschaft über ε-Umgebung definieren

$N_\varepsilon(y) := \{ x \mid Ax = b, x \geq 0, || y-x || \leq \varepsilon \}$

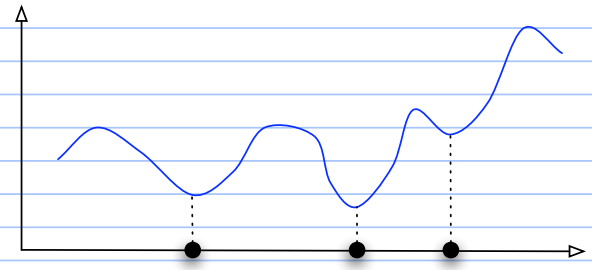


Lokale und globale Optima

Sei Problemklasse P_0 mit Nachbarschaft N gegeben und $I \in P_0$

$y \in S_I$ heißt lokal optimal bzgl. N, falls $c(y) \leq c(x)$ für alle $x \in N_I(y)$

2.7 Beispiel: Lokale Minima in der Analysis



2.8 Beispiel: TSP

Lokal optimale Lösungen bzgl. N_k heißen k-optimal

Exakte Nachbarschaft

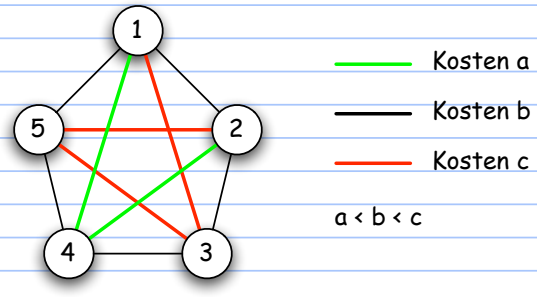
Eine Nachbarschaft bzgl. einer Problemklasse P_0 heißt exakt

:<=> für jede Instanz I gilt: ist $y \in S_I$ lokal optimal bzgl. N_I, so ist y global optimal für I

2.9 Beispiel: TSP

N_2 ist nicht exakt

Gegenbeispiel:



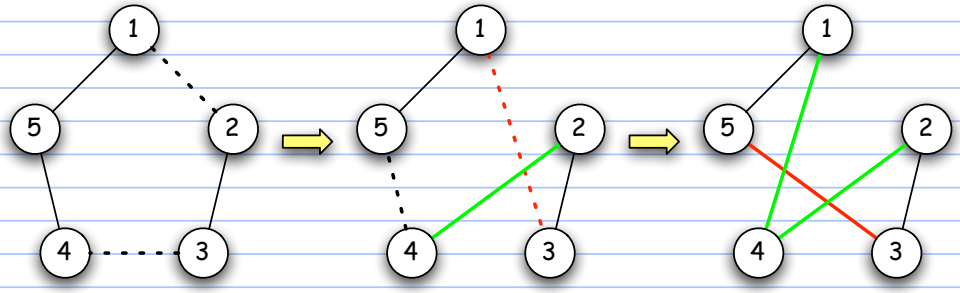
Tour y = "außen rum" hat Kosten 5b

Da die beiden grünen Kanten adjazent sind, kommt bei jedem 2-Tausch höchstens eine grüne Kante dazu, im günstigsten Fall also eine grüne und eine rote

Die neue Tour ist schlechter, falls $a+c > 2b$

=> y lokal optimal bzgl. N_2

Durch doppelten 2-Tausch können die beiden grünen Kanten in die Tour kommen



Diese Tour ist besser als y, falls $2a+c < 3b$

$a < b < c$ und $a+c > 2b$ und $2a+c < 3b$ ist erfüllt für $a = 1, b = 4, c = 8$

N_n ist exakt

klar, da $N_n(y) = S_I$ ❑

2.10 Beispiel: MST

Die Nachbarschaft von MST ist exakt

Beweis:

Nutze Satz aus ADM I:

T ist optimal <=> Für jede Nicht-Baum-Kante e gilt: e ist teuerste Kante in dem von e erzeugten Kreis bzgl. T

❑

Nachbarschaft motiviert das Prinzip der Lokalen Suche

Algorithmus Lokale Suche

Input

Instanz I eines Optimierungsproblems P_0 mit Nachbarschaft N_I

Startlösung $y \in S_I$

Output

lokales Optimum bzgl. N_I

Methode

iterative Verbesserung

while es gibt bessere Lösung $x \in N_I(y)$ do

wähle bessere Lösung $x \in N_I(y)$

y := x

return y

2.11 Satz (Lokale Suche für MST)

Die lokale Suche bzgl. der MST-Nachbarschaft ist ein polynomialer Algorithmus zur Bestimmung eines (global) optimalen MST, falls

(a) immer eine Nichtbaumkante f gewählt wird, die billiger als die teuerste Kante auf dem von f induzierten Kreis K ist

(b) immer die teuerste Baumkante e aus dem von f induzierten Kreis K entfernt wird

Beweis:

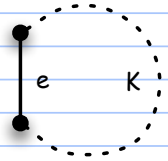
1. Da die Nachbarschaft exakt ist, liefert der Algorithmus bei Terminierung eine global optimale Lösung

2. Der Algorithmus terminiert in polynomialer Zeit

Claim 1: Eine ausgetauschte Kante kommt nie wieder zurück in den Baum

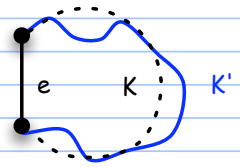
Beweis durch Widerspruch

Sei K der Kreis beim Austausch der Kante e, d.h. e wird entfernt.



Betrachte späteren Zeitpunkt t, zu dem e zum ersten Mal als Nichtbaumkante gewählt wird und wieder in den Baum kommt.

=> e erzeugt einen Kreis K'



=> K' ist aus K entstanden durch die Folge der lokalen Suchschritte bis zum Zeitpunkt t

In jedem dieser Schritte schließt e (als Nichtbaumkante) einen Kreis K(e) im momentanen Baum

Claim 2: In jedem Schritt gilt $c(e) \geq c(g)$ für alle Kanten $g \in K(e)$

Beweis durch Induktion entlang der Folge von Kreisen $K = K_1, K_2, K_3, \ldots$

Induktionsanfang (klar für $K = K_1$ nach Konstruktion)

Induktionsschritt von K_i auf K_{i+1}

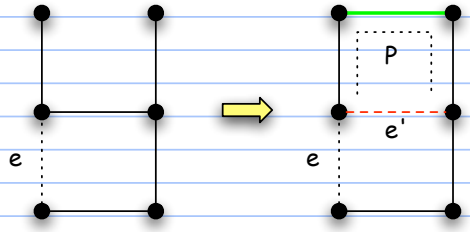
klar für $K_i = K_{i+1}$

sei $K_i \neq K_{i+1}$

Dann sind 2 Fälle möglich

K_{i+1} entsteht durch Vergrößerung von K_i

=> $K_{i+1} = K_i$ - (einige Kanten inklusive der momentan gelöschten Kante e') + P (P ist Teil des momentanen Kreises aus dem wir e' löschen)

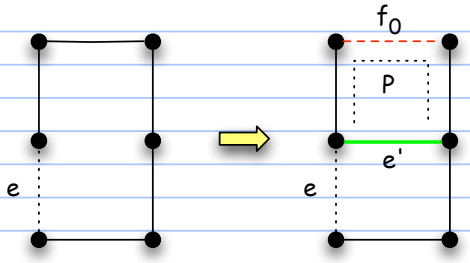


=> $c(e') \geq c(f)$ für alle Kanten $f \in P$ wegen (b)

$c(e) \geq c(e')$ nach Induktionsvoraussetzung, (b) => $c(e) \geq c(f)$ für alle Kanten $f \in K_{i+1}$

K_{i+1} entsteht durch Verkleinerung von K_i

=> $K_{i+1} = K$ + (einige Kanten inklusive der momentan hinzugefügten Kante e') - P (P ist Teil von K_i)



=> $c(e') < c(f_0)$ für entfernte Kante f_0, $c(e) \geq c(f)$ für alle Kanten $f \in P$ nach Induktionsvoraussetzung

=> $c(e) \geq c(f)$ für alle Kanten $f \in K_{i+1}$

Claim 2 => Widerspruch zur Wahl von e gemäß (a)

Also gibt es maximal m-n+1 Austauschschritte (m = # Kanten, n = # Knoten). Jeder dieser Schritte kann in $O(n)$ durchgeführt werden

teuerste Kante e im induzierten Kreis ermitteln und mit der Nichtbaumkante f vergleichen

[durch Breitensuche in O(# Kanten im Baum) = $O(n)$]

- e und f austauschen, falls c(e) < c(f)

  [bei Speicherung des Baumes als Array von Adjazenzlisten in $O(n)$]

- Insgesamt also $O((m-n+1)n) = O(mn)$ ❑

- Bemerkung: Haben in ADM I bessere Algorithmen kennengelernt, Kruskal $O(m \cdot \log n)$ und Prim $O(n^2)$

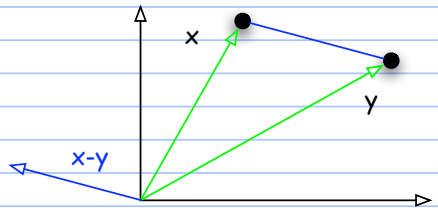
- Übung: Analyse der lokalen Suche für TSP mit der k-Opt Nachbarschaft. Ist sie polynomial?

- Konvexkombination zweier Vektoren
- Seien $x, y \in R^n$. Dann heißt jeder Punkt der Form

  $z = \lambda \cdot x + (1-\lambda) \cdot y$ mit $0 \leq \lambda \leq 1$

  eine Konvexkombination von x und y (strikte Konvexkombination falls $0 < \lambda < 1$)

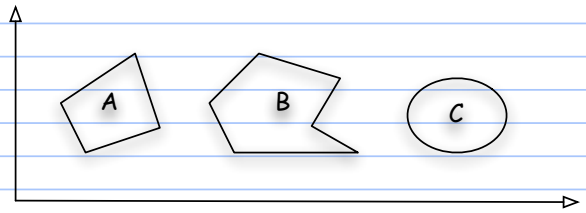
- die Konvexkombinationen sind genau die Punkte auf der Verbindungsgeraden zwischen x und y



Punkte auf Verbindungsgeraden haben als Vektoren die Form $y + \lambda(x-y)$

- Konvexe Menge
- $S \subseteq R^n$ heißt konvex, falls S alle Konvexkombinationen von je zwei Punkten $x, y \in S$ enthält
- 2.12 Beispiel:

- R^n, Ø, {x}, $x \in R^n$ sind konvex
- Im R^1 sind die konvexen Mengen genau die Intervalle
- Die konxen Mengen im R^2 sind solche ohne "Einbuchtungen"



2.13 Lemma

Der Durchschnitt von (beliebig vielen) konvexen Mengen ist konvex

Beweis:

Sei $S = \cap_{i \in I} S_i$, S_i konvex

Seien $x, y \in S$, $0 \le \lambda \le 1$, $z = \lambda \cdot x + (1-\lambda) \cdot y$

Def. S => $x,y \in S_i$ für alle i => $z \in S_i$ für alle i => $z \in S$ ❑

Lemma 2.13 ist Basis für die Definition der konvexen Hülle einer Menge

Die konvexe Hülle conv(S) einer Menge S ist die kleinste konvexe Menge, die S enthält, d.h.

$$conv(S) = \bigcap_{S \subseteq M,\ M \text{ konvex}} M$$

Dieser Durchschnitt existiert, da $M = R^n$ an der Durchschnittsbildung beteiligt ist

eine äquivalente Beschreibung (Übung) ist

$conv(S) = \{ \lambda_1 x^1 + \ldots + \lambda_k x^k \mid x^i \in S,\ \lambda_i \ge 0,\ \sum \lambda_i = 1,\ k \text{ endlich} \}$

Satz von Caratheodory: $k \le n+1$ reicht im R^n

Konvexe Funktion

Sei $S \subseteq R^n$ konvex. Eine Funktion $c : S \to R^1$ heißt konvex in S, falls

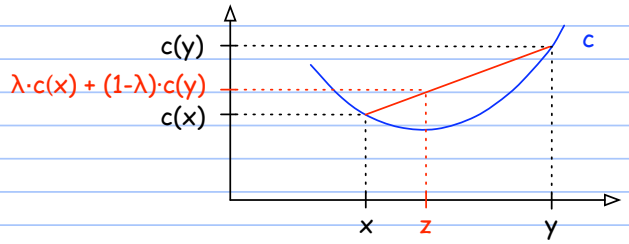
$$c(\lambda \cdot x + (1-\lambda) \cdot y) \le \lambda \cdot c(x) + (1-\lambda) \cdot c(y)$$

für alle $x, y \in S$, $0 \le \lambda \le 1$

2.14 Beispiel

Jede lineare Funktion ist konvex

Interpretation von konvexen Funktionen $c : R^1 \to R^1$



$z := \lambda \cdot x + (1-\lambda) \cdot y \qquad c(z) \leq \lambda \cdot c(x) + (1-\lambda) \cdot c(y)$

2.15 Lemma

Sei c konvex in $S \subseteq R^n$. Dann ist für jede reelle Zahl t die Niveaumenge

$$S_t := \{ x \in S \mid c(x) \leq t \}$$

konvex

Beweis:

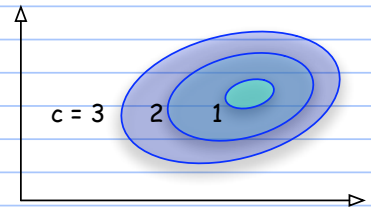
Sei $z := \lambda \cdot x + (1-\lambda) \cdot y$ mit $x, y \in S_t$, $0 \leq \lambda \leq 1$

$\Rightarrow$ $c(z) \leq \lambda \cdot c(x) + (1-\lambda) \cdot c(y)$ da c konvex

$\leq \lambda \cdot t + (1-\lambda) \cdot t$ da $x, y \in S_t$

$\Rightarrow$ $z \in S_t$ ❑

Niveaumengen einer konvexen Funktion $c : R^2 \rightarrow R^1$



Konkave Funktion

c in $S \subseteq R^n$ heißt konkav, falls -c konvex ist

$\Leftrightarrow$ $c(\lambda \cdot x + (1-\lambda) \cdot y) \geq \lambda \cdot c(x) + (1-\lambda) \cdot c(y)$ für alle $x, y \in S$, $0 \leq \lambda \leq 1$

Konvexe Optimierung = Minimierung einer konvexen Funktion über einer konvexen Menge. ...

2.16 Satz (lokal - global)

Sei I eine Instanz eines Optimierungsproblems mit $S_I \subseteq R^n$ konvex und c konvex in S_I.

=> Die Nachbarschaft

$$N_\varepsilon(y) := \{ x \in S_I : || y-x || \leq \varepsilon \}$$

definiert durch den Euklidischen Abstand ist für jedes $\varepsilon > 0$ exakt.

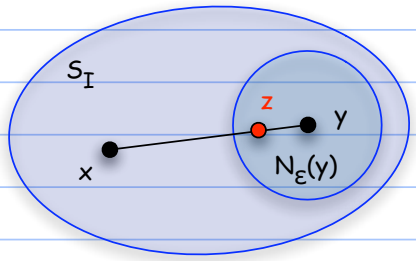
Beweis:

Sei $\varepsilon > 0$ fest und y lokales Optimum bzgl. N_ε.

Sei $x \in S_I$ beliebig. Zeige $c(y) \leq c(x)$. Klar falls $x \in N_\varepsilon(y)$

Sei also $x \notin N_\varepsilon(y)$

=> für geeignete Wahl von λ ist $z := \lambda \cdot x + (1-\lambda) \cdot y \in N_\varepsilon(y)$ und $z \neq y$.



c konvex => $c(z) \leq \lambda \cdot c(x) + (1-\lambda) \cdot c(y)$, außerdem $c(z) \geq c(y)$, da y lokal optimal

$$\Rightarrow c(x) \geq \frac{c(z) - (1-\lambda)c(y)}{\lambda} \geq \frac{c(y) - (1-\lambda)c(y)}{\lambda} = c(y)$$

Also $c(x) \geq c(y)$ ❑

Beachte: dies gilt ohne weitere Annahmen an c; insbesondere braucht c nicht differenzierbar zu sein.

Historische Definition konvexer Optimierungsprobleme

Eine Instanz I eines Optimierungsproblems heißt konvexes Optimierungsproblem falls

- S_I ist gegeben als Menge aller $x \in R^n$, die folgende Nebenbedingungen erfüllen:

  $g_i(x) \geq 0$ $\quad i = 1,\ldots,m$

  $g_i : R^n \to R^1$ konkav, $\quad i = 1,\ldots,m$

- c ist konvex in S_I

2.17 Lemma

Die zulässige Menge S_I einer Instanz I eines historisch definierten konvexen Optimierungsproblem ist konvex

Beweis:

g_i konkav $\Rightarrow$ $-g_i$ konvex

$\Rightarrow$ $S_i := \{ x \mid -g_i(x) \leq 0 \} = \{ x \mid g_i(x) \geq 0 \}$ konvex

$\Rightarrow$ $S_I = \cap_i S_i$ ist konvex wegen Lemma 2.13 ❑

2.18 Satz

In einem konvexen Optimierungsproblem ist jedes lokale Optimum ein globales Optimum

2.19 Bemerkung

- Es kann viele globale Optima geben
- Jede Instanz von LP ist ein konvexes Optimierungsproblem

  $\Rightarrow$ jedes lokale Minimum ist ein globales Minimum
- Die Analysis bietet hinreichende Kriterien für die Konvexität glatter Funktionen:
  - $D \subseteq R^n$ offen, $c : D \to R^1$ ist 2-mal stetig differenzierbar,

    Hessematrix (= Matrix der 2. partiellen Ableitungen) von c ist positiv semidefinit

3.1 Formen des Linearen Optimierungsproblem

(3.1) Allgemeine Form

$\min c^T x \qquad x \in \mathbb{R}^n, c \in \mathbb{R}^n$

unter

$a_i^T x$	$=$	b_i	$i \in M$	$a_i \in \mathbb{R}^n$
$a_i^T x$	$\geq$	b_i	$i \in \overline{M}$	
x_j	$\geq$	0	$j \in N$	
x_j		beliebig	$j \in \overline{N}$	

3.1 Beispiel: Diät-Problem (historisch ältestes LP)

n Nahrungsmittel, j = 1,..., n

m Bestandteile (Eiweiße, Vitamine usw.) i = 1,..., m

a_{ij} = Anteil von Bestandteil i in Einheit von Nahrungsmittel j,

r_i = erforderlicher Anteil von Bestandmittel i pro Zeitraum (Woche)

x_j = Verbrauch an Nahrungsmittel j pro Zeitraum (Woche)

c_j = Kosten pro Einheit von Nahrungsmittel j

$x = (x_1, x_2, ..., x_n)^T$ entspricht wöchentlicher Diät

zulässige Diät entspricht $Ax \geq r$ mit $A = (a_{ij})$, $r = (r_1, r_2, ..., r_m)^T$

(3.2) Ermittlung einer "billigsten" zulässigen Diät ist LP

$\min\ c^T x$

unter $Ax \geq r$

$x \geq 0$

- Ein LP in der Form (3.2) heißt in kanonischer Form
- Ein LP in der Form ...
- Ein LP in der Form (3.1) heißt in allgemeiner Form ...
- 3.2 Lemma (Äquivalenz der drei Formen)
  - Alle 3 Formen sind äquivalent
    - in dem Sinne, dass eine Instanz I der einen Form durch eine einfache Transformation in eine Instanz I' einer anderen Form transformiert werden kann, so dass aus einer optimalen Lösung von I' einfach eine optimale Lösung von I konstruiert werden kann
  - Beweis
    - es reicht die Angabe folgender Transformationen:
      - allgemeine Form -> kanonische Form
        - eliminiere Gleichheitsrestriktionen und unbeschränkte Variable

- $\sum_{j=1}^n a_{ij}x_j = b_i \quad \rightarrow \quad \sum_{j=1}^n a_{ij}x_j \geq b_i$ und $\sum_{j=1}^n a_{ij}x_j \leq b_i$
- x_j unbeschränkt $\quad \rightarrow \quad x_j = x_j^+ - x_j^-, \quad x_j^+, x_j^- \geq 0$

- allgemeine Form -> Standardform
  - "$\geq$" eliminieren durch Überschussvariable $s_i \geq 0$
    $\sum_{j=1}^n a_{ij}x_j \geq b_i \quad \rightarrow \quad \sum_{j=1}^n a_{ij}x_j - s_i = b_i$
  - "$\leq$" eliminieren durch Schlupfvariable $s_i \geq 0$
    $\sum_{j=1}^n a_{ij}x_j \leq b_i \quad \rightarrow \quad \sum_{j=1}^n a_{ij}x_j + s_i = b_i$ ❑

Ziel: Entwicklung eines Algorithmus zur Lösung von LP

Ausgangspunkt: LP in Standardform

$$\min\ c^T x$$

$$\text{unter}\quad Ax = b$$

$$x \geq 0$$

mit Zusatzannahmen 3.1 - 3.3 (werden uns später davon befreien)

Annahme 3.1: A ist (mxn)-Matrix mit vollem Zeilenrang m

3.3 Beispiel

$$\begin{array}{lrcrcrcrcl}
\min & 2x_1 & & & & +x_4 & & +5x_7 & & \\
\text{unter} & x_1 & +x_2 & +x_3 & +x_4 & & & & = & 4 \\
& x_1 & & & & +x_5 & & & = & 2 \\
& & & +x_3 & & & +x_6 & & = & 3 \\
& & +3x_2 & +x_3 & & & & +x_7 & = & 6 \\
& x_j & \geq 0 & \forall j & & & & & &
\end{array}$$

$$(A|b) = \left(\begin{array}{cccccc|c}
1 & 1 & 1 & 1 & & & 4 \\
1 & & & & 1 & & 2 \\
 & & 1 & & & 1 & 3 \\
 & 3 & 1 & & & & 6
\end{array}\right)$$

A hat vollen Zeilenrang $m = 4$

Konsequenzen aus der Linearen Algebra und daraus abgeleitete Definitionen:

Spaltenrang = Zeilenrang = rang(A) = m

Eine Auswahl von m linear unabhängigen Spalten $B = \{A_j, A_k, \ldots, A_r\}$ von A heißt Basis, und die zugehörigen Spalten heißen Basisspalten. Die anderen Spalten heißen Nichtbasisspalten.

Mit B (gelegentlich A_B) bezeichnen wir auch die Teilmatrix dieser Spalten und die zugehörigen Indizes mit $B(1),\ldots, B(m)$,

also $B = (A_{B(1)}, A_{B(2)}, \ldots, A_{B(m)})$. Gelegentlich identifizieren wir B mit der Menge der Indizes, also $B = \{ B(1), \ldots, B(m) \}$.

Mit A_N bezeichnen wir die Teilmatrix der Nichtbasisspalten.

A hat maximal $\binom{n}{m}$ Basen

3.3 Beispiel (Fortsetzung)

Basis 1:

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 4 \\ 1 & & & 1 & 2 \\ & & 1 & & 3 \\ & 3 & 1 & & 6 \end{array}\right)$$

B(1) = 4, B(2) = 5, B(3) = 6, B(4) = 7

Basis 2

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 4 \\ 1 & & & 1 & 2 \\ & & 1 & & 3 \\ & 3 & 1 & & 6 \end{array}\right)$$

B(1) = 2, B(2) = 5, B(3) = 6, B(4) = 7

Basis 3

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 4 \\ 1 & & & 1 & 2 \\ & & 1 & & 3 \\ & 3 & 1 & & 6 \end{array}\right)$$

B(1) = 2, B(2) = 1, B(3) = 3, B(4) = 7

$$B = \begin{pmatrix} 1 & 1 & 1 & \\ & 1 & & \\ & & & 1 \\ 3 & & 1 & 1 \end{pmatrix}$$

Jede Basis ist invertierbar und kann mit elementaren Zeilenoperationen und Zeilenpermutationen (Gauß-Algorithmus) auf die Einheitsmatrix transformiert werden.

Transformiert man die ganze erweiterte Matrix (A|b) mit diesen Operationen, so bekommt man eine Lösung von Ax = b, indem man die Basisvariablen auf die (transformierte) rechte Seite setzt, und die Nichtbasisvariablen auf 0 setzt. Diese Lösung heißt Basislösung zur Basis B.

Nützlich dafür ist folgendes Applet

http://people.hofstra.edu/faculty/Stefan_Waner/RealWorld/tutorialsf1/scriptpivot2.html

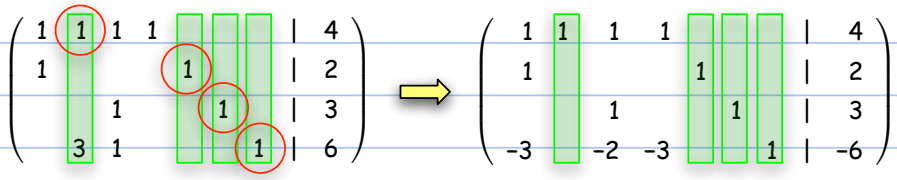
3.3 Beispiel (Fortsetzung)

Basis 1

keine Transformation nötig, da B = Einheitsmatrix

Basislösung dazu: $x_4 = 4, x_5 = 2, x_6 = 3, x_7 = 6, x_j = 0$ sonst

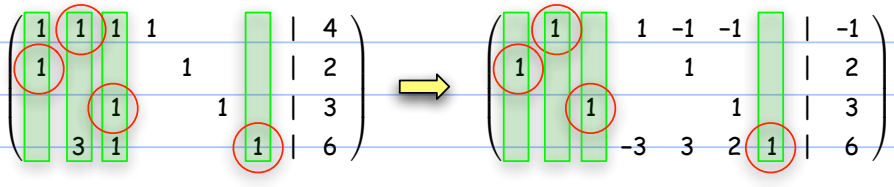
Basis 2



B(1) = 2, B(2) = 5, B(3) = 6, B(4) = 7

Basislösung dazu: $x_2 = 4, x_5 = 2, x_6 = 3, x_7 = -6, x_j = 0$ sonst

Basis 3



B(1) = 2, B(2) = 1, B(3) = 3, B(4) = 7

Basislösung dazu: $x_2 = -1, x_1 = 2, x_3 = 3, x_7 = 6, x_j = 0$ sonst

Permutiert man die Spalten in A und x so, dass $A = (A_B, A_N)$ und $x = (x_B, x_N)^T$, so entsprechen die elementaren Umformungen der Multiplikation von $(A_B, A_N)\,(x_B, x_N)^T = b$ mit der Basisinversen B^{-1} von links:

$$B^{-1}(A_B, A_N)\,(x_B, x_N)^T = B^{-1}b$$

$$\Leftrightarrow \quad B^{-1}A_B x_B + B^{-1}A_N x_N = B^{-1}b$$

$$\Leftrightarrow \quad x_B + B^{-1}A_N x_N = B^{-1}b$$

Setzen wir in der Basislösung $x_N = 0$, so folgt $x_B = B^{-1}b$

Also: Ist B eine Basis von A, so ergibt sich die zugehörige Basislösung $x = (x_B, x_N)^T$ als

$x_B = B^{-1}b, \; x_N = 0$

- 3.3 Beispiel (Fortsetzung)
  - Basis 3

$$B = \begin{pmatrix} 1 & 1 & 1 & \\ & 1 & & \\ & & 1 & \\ 3 & & 1 & 1 \end{pmatrix} \Rightarrow B^{-1} = \begin{pmatrix} 1 & -1 & -1 & \\ & 1 & & \\ & & 1 & \\ -3 & 3 & 2 & 1 \end{pmatrix}$$

$$\Rightarrow x_B = \begin{pmatrix} x_2 \\ x_1 \\ x_3 \\ x_7 \end{pmatrix} = B^{-1}b = \begin{pmatrix} 1 & -1 & -1 & \\ & 1 & & \\ & & 1 & \\ -3 & 3 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} 4 \\ 2 \\ 3 \\ 6 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \\ 6 \end{pmatrix}$$

- Eine Basislösung x heißt zulässige Basislösung falls $x \geq 0$, d.h. x ist zulässige Lösung des LP
- 3.3 Beispiel (Fortsetzung)
  - Die Basislösung zu Basis 1 ist zulässig, die zu Basis 2 und Basis 3 sind es nicht.

- **Rolle der Basislösungen für den Simplex-Algorithmus**

- Der Simplexalgorithmus wird sich aus algebraischer Sicht als lokale Suche auf der Menge der zulässigen Basislösungen herausstellen
- Brauchen dazu noch
  - Umgebungsbegriff (zwei zulässige Basislösungen sind benachbart, wenn sie sich in genau einer Spalte der Matrix A unterscheiden)
  - Algorithmische Analyse des Übergangs von einer zulässigen Basislösung zur nächsten (Pivotoperation, Pivotschritt)
  - Vorher jedoch einige mathematische Aussagen über (zulässige) Basislösungen

- **Einige mathematische Eigenschaften (zulässiger) Basislösungen**
  - 3.4 Lemma (Die Werte von Basisvariablen sind beschränkt)
    - Die Einträge von A und b seien ganzzahlig und x sei eine Basislösung von $Ax = b$.
      Sei $\alpha := \max_{ij} |a_{ij}|$ und $\beta := \max_i |b_i|$. Dann gilt
      $|x_j| \leq m! \, \alpha^{m-1} \, \beta$ für alle j
    - Beweis:
      - Sei $x_B = B^{-1}b$ mit $B = (A_{B(1)}, A_{B(2)}, \ldots, A_{B(m)})$.

Nach der Cramerschen Regel ist

$$x_{B(i)} = \frac{\det B^i}{\det B} \quad \text{mit } B^i = (A_{B(1)}, \ldots, A_{B(i-1)}, b, A_{B(i+1)}, \ldots, A_{B(m)})$$

- A, b ganzzahlig => det B ganzzahlig => $|\det B| \geq 1$
- $\det B^i$ nach Spalte b entwickeln ergibt m Summanden der Form $b_i \cdot$ [(m-1)x(m-1) Unter-Determinante von A], jede solche Unterdeterminante ist Summe von (m-1)! Produkten von (m-1) Einträgen aus A. Also:
- $$|x_{B(i)}| = \frac{|\det B^i|}{|\det B|} \leq |\det B^i| \leq m \cdot \beta \cdot (m-1)! \cdot a^{m-1} = m! a^{m-1} \beta$$ ❑

3.5 Lemma (Man kann auf keine zulässige Basislösung verzichten, jede kann optimal sein)

Sei x eine zuässige Basislösung von $Ax = b$, $x \geq 0$ zur Basis B.

=> es existiert ein Kostenvektor c so dass x die einzige Optimallösung von

$$\min c^T x$$
$$\text{unter } Ax = b$$
$$x \geq 0$$

ist.

Beweis:

Setze $c_j := \begin{cases} 0 & \text{falls } j \in B \\ 1 & \text{falls } j \notin B \end{cases}$

=> $c^T x = 0$, da Nichtbasisvariablen 0 sind

=> x ist optimal, da $c^T y \geq 0$ für jede zulässige Lösung y

sei y weitere Optimallösung

=> $y_j = 0$ für alle $j \notin B$

=> $Ay = b$ reduziert sich zu $By_B = b$ => $y_B = B^{-1} b = x_B$

=> x ist einzige Optimallösung ❑

Werden später zeigen, dass zulässige Basislösungen auch ausreichen, d.h. sie bilden die kleinste Menge zulässiger Lösungen auf der das Optimum für alle Kostenfunktionen c angenommen wird

Existenz zulässiger Basislösungen

Frage: besitzt jedes LP eine zulässige Basislösung?

Annahme 3.2: Der zulässige Bereich S_I des LP sei nichtleer

3.6 Satz (Existenz von zulässigen Basislösungen)

Unter den Annahmen

3.1: rang(A) = m

3.2: $S_I \neq \emptyset$

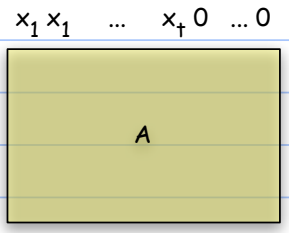
existiert mindestens eine zulässige Basislösung

Beweis:

$S_I \neq \emptyset$ => es gibt Lösungen

sei x eine Lösung mit den meisten Null-Komponenten und o.B.d.A. $x_1, \ldots, x_t > 0$, $x_{t+1}, \ldots, x_n = 0$

=> Ax = b reduziert sich zu $A_1x_1 + \ldots + A_tx_t = b$ (3.4)



Sei $A' := (A_1, \ldots, A_t)$ und $r := \text{rang}(A')$

=> $0 \leq r \leq \min\{t, m\}$

Fallunterscheidung

r = 0

=> $A_j = 0$ für $j = 1, \ldots, t$ => (3.4) Ax = 0

=> (Wahl von x) x = 0

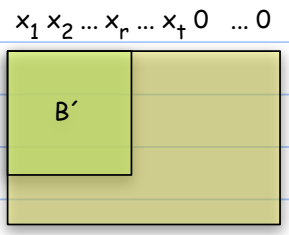
=> x ist zulässige Basislösung zu jeder Basis, eine Basis existiert wegen Annahme 3.1

$0 < r < t$

erzeugen einen Widerspruch

sei B´ eine nicht-singuläre Teilmatrix von A' mit Rang r, o.B.d.A

$$B' = \begin{pmatrix} a_{11} & \ldots & a_{1r} \\ \vdots & \ddots & \vdots \\ a_{r1} & \ldots & a_{rr} \end{pmatrix}$$



sei B_j' die Spalte der ersten r Komponenten von A_j (j = 1, ..., t) und b´ der Vektor der ersten r Komponenten von b

r < t => die letzten m - r Gleichungen von (3.4) sind redundant und wir können (3.4) schreiben als

$B_1' x_1 + ... + B_t' x_t = b'$

<=> $B_1' x_1 + ... + B_r' x_r = b' - (B_{r+1}' x_{r+1} + ... + B_t' x_t)$

$$\Leftrightarrow \quad (B_1', \ldots, B_r') \begin{pmatrix} x_1 \\ \vdots \\ x_r \end{pmatrix} = b' - (B_{r+1}', \ldots, B_t') \begin{pmatrix} x_{r+1} \\ \vdots \\ x_t \end{pmatrix}$$

Multiplikation von links mit $(B')^{-1}$ ergibt

$$\begin{pmatrix} x_1 \\ \vdots \\ x_r \end{pmatrix} = (B')^{-1} b' - (B')^{-1} (B_{r+1}', \ldots, B_t') \begin{pmatrix} x_{r+1} \\ \vdots \\ x_t \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} x_1 \\ \vdots \\ x_r \end{pmatrix} = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_r \end{pmatrix} + \begin{pmatrix} \alpha_{1,r+1} & \cdots & \alpha_{1t} \\ \vdots & \ddots & \vdots \\ \alpha_{r,r+1} & \cdots & \alpha_{rt} \end{pmatrix} \begin{pmatrix} x_{r+1} \\ \vdots \\ x_t \end{pmatrix} \quad (*)$$

für Zahlen α_{ij} und β_i ,

d.h., $x_1, ..., x_r$ ergeben sich affin linear aus $x_{r+1}, ..., x_t$

Nutzen (*) um eine neue zulässige Lösung mit mehr Null-Komponenten zu konstruieren

es gibt Zeile i mit $\alpha_{i,r+1} \neq 0$ (sonst könnte $x_{r+1} = 0$ gewählt werden im Widerspruch zur Wahl von x)

variiere x_{r+1} so dass eines der $x_1, ..., x_r$ oder x_{r+1} Null wird, aber alle ≥ 0 bleiben

$\alpha_{i,r+1} < 0$ => x_i wächst falls x_{r+1} kleiner wird => $x_1, ..., x_t$ bleiben ≥ 0

$\alpha_{i,r+1} > 0$ => Verkleinern von x_{r+1} um $\theta > 0$ ergibt die Bedingung $x_i - \alpha_{i,r+1} \cdot \theta \geq 0$

d.h. $\theta \leq \dfrac{x_i}{\alpha_{i,r+1}}$

wähle $\theta := \min \{ x_{r+1}, \dfrac{x_i}{\alpha_{i,r+1}} \mid \alpha_{i,r+1} > 0 \}$

und setze $y_{r+1} := x_{r+1} - \theta$ und $y_j := x_j$ für $j > r+1$

=> $y_1, ..., y_r$ ergeben sich gemäß (*) und sind ≥ 0

=> $y = (y_1, ..., y_n)^T$ ist zulässige Lösung des LP

2 Möglichkeiten:

- $y_{r+1} = 0$ (tritt ein wenn $\theta = x_{r+1}$)
- oder ein $y_1, \ldots, y_r$ wird 0
- => Widerspruch zur Wahl von x

$0 < r = t \leq m$

A' hat t Spalten, $\text{rang}(A') = t$ => die t Spalten von A' sind linear unabhängig

=> $(\text{rang}(A) = m)$ sie können ggf. durch Hinzunahme von Spalten aus A zu einer Basis B von A ergänzt werden

=> x ist zulässige Basislösung zu B ❑

Beschränktheit des Lösungsraums

Letzte Annahme

Annahme 3.3: $\{ c^T x \mid Ax = b, x \geq 0 \}$ ist nach unten beschränkt

werden zeigen, dass dannn schon der Zulässigkeitsbereich $S_I = \{ x \mid Ax = b, x \geq 0 \}$ als beschränkt angenommen werden kann

3.7 Satz (Der zulässige Bereich kann als beschränkt angenommen werden)

Es gelten die Annahmen

3.1: $\text{rang}(A) = m$

3.2: $S_I \neq \emptyset$

3.3: $\{ c^T x \mid Ax = b, x \geq 0 \}$ ist nach unten beschränkt

Dann ist das LP

$$\min \{ c^T x \mid Ax = b, \ x \geq 0 \} \qquad \text{(LP)}$$

äquivalent zu dem LP

$$\min \{ c^T x \mid Ax = b, \ x \geq 0, \ x_j \leq M \} \qquad \text{(LP*)}$$

mit $M := (m+1)! \, \alpha^m \, \beta$

$\alpha := \max_{ij} \{ |a_{ij}|, |c_j| \}$

$\beta := \max_i \{ |b_i|, |z| \}$

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

in dem Sinne, dass die Optimalwerte übereinstimmen und (LP) und (LP*) mindestens eine gemeinsame Optimallösung haben, die Basislösung von (LP) ist.

Beweis:

Sei $G := \{ c^T x \mid Ax = b, \ x \geq 0 \} \subseteq R^1$ => $z := \inf G < \infty$ wegen Annahme (3.3)

G ist abgeschlossen (da über = bzw. $\geq$ und lineare Abbildung definiert) => $z \in G$

=> $\{ x \in R^n \mid c^T x = z,\ Ax = b,\ x \geq 0 \}$ ist Menge der Optimallösungen von (LP) (3.5)

Zwei Fälle

$\text{rang}\{ c^T x = z,\ Ax = b \} = m+1$

Satz 3.6 => (3.5) hat eine zulässige Basislösung x mit Basis B

Lemma 3.4 => x erfüllt die Schranken $x_j \leq M$

Außerdem: x ist zulässig für (LP)

(3.5) => x ist optimal für (LP) und damit auch für (LP*), da $S_{LP^*} \subseteq S_{LP}$

rang = m+1 => B ohne Zeile zu $c^T x = z$ enthält m unabhängige Spalten aus A

=> x ist Basislösung von (LP)

$\text{rang}\{ c^T x = z,\ Ax = b \} = m$

=> c ist Linearkombination der Zeilen a_i von A, etwa $c = \sum d_i a_i$

=> $c^T x = (\sum d_i a_i)^T x = \sum d_i a_i^T x = \sum d_i b_i$ = konstant, unabhängig von x

=> jede Lösung von (LP) ist Optimallösung, insbesondere eine Basislösung.

Diese erfüllt die Schranken wegen Lemma 3.4 ❑

Hauptausagen dieses Abschnittes

Der zulässige Bereich eines LP in kanonischer Form ist ein Polyeder

Die zulässigen Basislösungen des zugehörigen LP in Standardform entsprechen den Ecken des Polyeders

Das Optimum wird in einer Ecke bzw. zulässigen Basislösung angenommen

Geometrische Grundlagen

$\emptyset \neq S \subseteq R^d$ ist linearer Teilraum von R^d

<=> S ist abgeschlossen unter Vektoraddition und skalarer Multiplikation

<=> S ist Lösungsmenge eines homogenen linearen Gleichungssystem $Ax = 0$

Dann ist $\dim(S) + \text{rang}(A) = d$

$T \subseteq R^d$ ist affiner Teilraum von R^d

<=> T ist ein um einen Vektor translatierter linearer Teilraum, d.h.

$T = \{ u + x \mid x \in S \}$, S linearer Teilraum, $u \in R^d$

<=> T ist Lösungsmenge eines inhomogenen linearen Gleichungssystem $Ax = b$

Dimension $\dim(T) := \dim(S)$

Dimension einer Menge $S \subseteq R^d$

= Dimension des kleinsten affinen Teilraums, der S enthält (affine Hülle von S)

Beispiele:

eine Gerade hat Dimension 1

$|S| = k \leq d+1 \Rightarrow \dim(S) \leq k-1$

Für die Lösungsmenge $S = \{ x \mid Ax = b, x \geq 0 \}$ eines LP in Standardform gilt $\dim(S) \leq n-m$

denn:

$\text{rang}(A) = m \Rightarrow \{ x \mid Ax = b \}$ hat Dimension $n-m$ (sofern $Ax = b$ lösbar)

die Vorzeichenbeschränkungen $x_j \geq 0$ können diese erniedrigen

z.B. wenn $\{ x \mid Ax = b \} \cap \{ x_j \geq 0 \} = \{0\}$

Hyperebene im R^d

= affiner Teilraum der Dimension $d-1$

= Lösungsmenge einer Gleichung $a_1x_1 + \ldots + a_dx_d = b$ (nicht alle $a_j = 0$)

Sie definiert zwei (abgeschlossene) Halbräume

$\{ x \mid a_1x_1 + \ldots + a_dx_d \geq b \}$ und $\{ x \mid a_1x_1 + \ldots + a_dx_d \leq b \}$

Polyeder im R^d

= nichtleerer Durchschnitt von endlich vielen Halbräumen

$\Rightarrow$ Polyeder sind konvex

Polytop im R^d

= beschränktes Polyeder

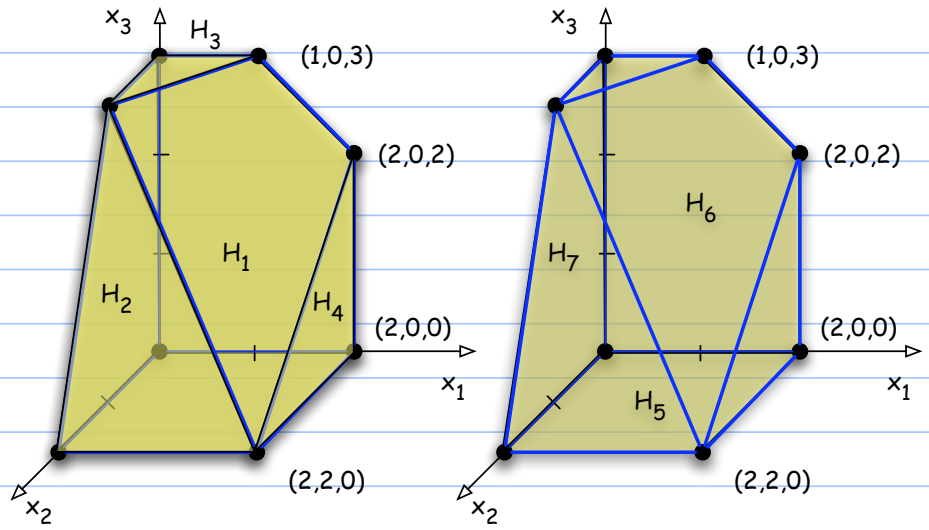
Beispiel: Platonische Körper

@ http://www.3quarks.com/GIF-Animations/PlatonicSolids/index-de.html

Geometrische Aspekte von Polytopen

Zulässigkeitsbereiche S von LPs in kanonischer Form sind Polyeder. Sie sind Polytope, falls S beschränkt ist bzw. als beschränkt angenommen werden kann.

3.8 Beispiel



Polytop ist Durchschnitt der Halbräume

$x_1 + x_2 + x_3 \leq 4$ Hyperebene H_1

$x_1 \leq 2$ Hyperebene H_4

$x_3 \leq 3$ Hyperebene H_3

$3x_2 + x_3 \leq 6$ Hyperebene H_2

$x_j \geq 0$ Hyperebenen H_5, H_6, H_7

Eine Hyperebene H berührt das Polyeder P

:<=> $H \cap P \neq \emptyset$ und P ist ganz in einem Halbraum zu H enthalten

$f := H \cap P$ heißt dann Seite oder Stützfläche von P (Englisch: face)

wichtig: Facette := Seite der Dimension d-1 (Englisch: facet)

Ecke := Seite der Dimension 0 (ein Punkt)

Kante := Seite der Dimension 1 (ein Geradenstück)

Einige geometrische Fakten (ohne Beweis)

(a) Die zu einer Facette gehörende Hyperebene H gehört zu einem P definierenden Halbraum (d.h. das Weglassen von H verändert P)

(b) Hyperebenen von beliebigen Seiten erfüllen i.A. nicht (a)

Halbraum $x_2 \leq 2$ in Beispiel 3.8 ist Seite ($\{ x_2 \leq 2 \} \cap P$) ist Kante, keine Facette)

Dieser Halbraum ist redundant, die zugehörige Ungleichung wird bereits durch andere impliziert:

$x_3 \geq 0$, $3x_2 + x_3 \leq 6$ => $3x_2 \leq 3x_2 + x_3 \leq 6$ => $x_2 \leq 2$

(c) Eine Kante ist eine Strecke, die zwei Ecken verbindet und auf dem Rand des Polyeders liegt (die Umkehrung gilt nicht)

3.9 Satz (Minkowski 1896)

(a) Jedes Polytop ist gleich der konvexen Hülle seiner Ecken,

d.h. jeder Punkt x eines Polytops P ist darstellbar als

$x = \lambda_1 x^1 + \ldots + \lambda_k x^k$, x^i Ecke von P, $\lambda_i \geq 0$, $\sum \lambda_i = 1$, k endlich

(b) Ist $V \subseteq R^d$ eine endliche Menge, so ist conv(V) ein Polytop P und $\{$ Ecken von P $\} \subseteq V$

Beweis

zum Teil Übung

(a) durch Induktion nach Dimension d

In Beispiel 3.8 ist

$$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix} + \frac{1}{3}\begin{pmatrix} 0 \\ 0 \\ 3 \end{pmatrix} + \frac{1}{6}\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

(b) ist schwieriger algebraisch zu beweisen, ist geometrisch intuitiv klar ❑

Folgerung: Können ein Polytop bereits auf 2 Weisen auffassen:

1. als konvexe Hülle einer endlichen Menge von Punkten

(z.B. der Inzidenzvektoren von Lösungen eines kombinatorischen Optimierungsproblems)

2. als Durchschnitt von endlich vielen Halbräumen (falls beschränkt)

(natürliche Art, wenn die Ungleichungen explizit gegeben sind, wie z.B. bei LPs in kanonischer Form)

eine dritte, algebraische, werden wir jetzt herleiten. Sie besteht aus einem lineares Gleichungssystem $Ax = b$, $x \geq 0$ dessen zulässige Basislösungen den Ecken von P entsprechen

Algebraische Interpretation von Polytopen

Vom Polytop zum Gleichungssystem

Sei P ein Polytop im R_+^{n-m} beschrieben durch die n Ungleichungen (3.6)

$$x_i \geq 0 \qquad i = 1, \ldots, n-m$$

$$h_{i,1}x_1 + \ldots + h_{i,n-m}x_{n-m} + g_i \leq 0 \qquad i = n-m+1, \ldots, n$$

Sei H die Koeffizientenmatrix der Restiktionen $i = n-m+1, \ldots, n$

Führe Schlupfvariablen x_i für die Restriktionen $i = n-m+1, \ldots, n$ ein (m Stück)

=> (3.6) wird zu $Ax = b,\ x \geq 0,\ x \in R^n$ mit $A = (H|I),\ b = -(g_{n-m+1}, \ldots, g_n)^T$

also zum zulässigen Bereich S eines LP in Standardform

Auf P definiert dies folgende Abbildung (Transformation) $f : P \to S$ (3.7)

$$x' = \begin{pmatrix} x'_1 \\ \vdots \\ x'_{n-m} \end{pmatrix} \in P \quad \mapsto \quad \begin{pmatrix} x_1 \\ \vdots \\ x_{n-m} \\ \vdots \\ x_n \end{pmatrix} \in S$$

$$\text{mit} \quad x_i = \begin{cases} x'_i & i = 1, \ldots, n-m \\ \underbrace{-g_i - \sum_{j=1}^{n-m} h_{ij} x'_j}_{\text{Schlupf}} & i = n-m+1, \ldots, n \end{cases}$$

Beachte: x ist eindeutig durch x' bestimmt, d.h. f ist injektiv

Vom Gleichungssystem zum Polytop

Sei $Ax = b,\ x \geq 0,\ x \in R^n$ der zulässige Bereich S eines LP in Standardform (mit den Voraussetzungen 3.1 - 3.3).

Dann existiert eine Basis B, und wir betrachten die Aufteilung $A = (A_B, A_N)$ und $x = (x_B, x_N)^T$ in Basis- und Nichtbasisvariablen und o.B.d.A. sei $B = \{ n-m+1, \ldots, n \}$ (letzte m Spalten von A)

Multiplikation von $Ax = b$ von links mit B^{-1} ergibt

$B^{-1} (A_B, A_N) (x_B, x_N)^T = x_B + B^{-1} A_N x_N = B^{-1} b =: b'$

Mit $B^{-1} A_j =: A'_j$ folgt

$$x_{B(i)} = b'_i - \sum_{j=1}^{n-m} a'_{ij} x_j \quad i = 1, \ldots, m$$

Also ist $Ax = b,\ x \geq 0$ äquivalent zu (3.8)

$$\begin{aligned} b'_i - \textstyle\sum_{j=1}^{n-m} a'_{ij} x_j &\geq 0 \quad i = 1, \ldots, m \\ x_j &\geq 0 \quad j \notin B \end{aligned}$$

Dies sind n Ungleichungen in den Variablen $x_1, \ldots, x_{n-m}$ die (da S beschränkt) ein Polytop P im R^{n-m} definieren

Auf S definiert dies folgende Abbildung $g : S \to P$ (für $B = \{ n-m+1, \ldots, n \}$) (3.9)

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_{n-m} \\ \vdots \\ x_n \end{pmatrix} \in S \quad \mapsto \quad x' = \begin{pmatrix} x'_1 \\ \vdots \\ x'_{n-m} \end{pmatrix} \in P$$

mit $x_j' := x_j \quad j = 1, \ldots, n-m$ (Schlupf in Basisvariablen wird "vergessen").

Offenbar ist g eine lineare Abbildung und injektiv, da $x_{n-m+1}, \ldots, x_n$ eindeutig durch $x_1, \ldots, x_{n-m}$ festgelegt sind.

Ferner gilt:

$x_j = 0$ für $j = 1, \ldots, n-m$, also $j \notin B$, bedeutet, dass x' in P auf einer Kordinatenhyperebene liegt

$x_j = 0$ für $j = n-m+1, \ldots, n$, also $j \in B$, bedeutet, dass eine P definierende Ungleichung mit Gleichheit erfüllt ist

(Basisvariablen "entsprechen" Schlupfvariablen, wobei diese Rolle willkürlich über die Wahl der Basis festgelegt wird)

3.8 Beispiel (Fortsetzung)

$$A = \left(\begin{array}{ccc:c:ccc} 1 & 1 & 1 & \vdots & 1 & & \\ 1 & & & \vdots & & 1 & \\ & & 1 & \vdots & & & 1 \\ & 3 & 1 & \vdots & & & 1 \end{array}\right) \qquad b = \begin{pmatrix} 4 \\ 2 \\ 3 \\ 6 \end{pmatrix}$$

$$x' = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \in P \quad \leftrightarrow \quad x = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 2 \\ 1 \\ 1 \\ 3 \end{pmatrix} \in S$$

Werte der eigentlichen Variablen

Werte der Schlupf-variablen

Beachte: f und g sind Bijektionen zwischen P und S, wobei g von der Wahl der Basis abhängt und P bestimmt.

3.10 Satz (Interpretation der Ecken in den drei Sichten)

Sei P ein Polytop und $S = \{ x \mid Ax = b, x \geq 0 \}$ die zugehörige zulässige Menge eines LP in Standardform.

Sei $y' = (y_1', ..., y_{n-m}')^T \in P$ und $y = (y_1, ..., y_n)^T \in S$ der zugehörige Vektor gemäß (3.7)

Dann sind folgende Aussagen äquivalent:

(1) y' ist eine Ecke von P

(2) y' ist nicht als strikte Konvexkombination anderer Punkte aus P darstellbar

(3) y ist zulässige Basislösung von S

Beweis

(1) => (2)

Annahme es gibt $z', z'' \in P$, $0 < \lambda < 1$ mit $y' = \lambda z' + (1-\lambda)z''$

y' Ecke => es gibt Halbraum $HR = \{ z \mid h^T z \leq g \}$ mit $HR \cap P = \{y'\}$

$z', z'' \notin HR$ => $h^T z' > g$ und $h^T z'' > g$ => $h^T(\lambda z' + (1-\lambda)z'') > g$

=> $y' \notin HR$, Widerspruch

(2) => (3)

Sei $y \in S$ gemäß (3.7) aus y' konstruiert. Sei $B' := \{ j \mid y_j > 0 \}$

Claim: Die Spalten A_j mit $j \in B'$ sind linear unabhängig

falls nicht, so existieren Zahlen d_j (nicht alle 0) mit $\sum_{j \in B'} d_j A_j = 0$ (3.10)

Definition von B' => $\sum_{j \in B'} y_j A_j = b$ (3.11)

(3.11) + θ·(3.10) sowie (3.11) - θ·(3.10) ergibt

$\sum_{j \in B'} (y_j + \theta d_j) A_j = b$ und $\sum_{j \in B'} (y_j - \theta d_j) A_j = b$

$y_j > 0$ für $j \in B'$ => $\theta > 0$ so wählbar, dass $y_j + \theta d_j \geq 0$ und $y_j - \theta d_j \geq 0$

dazu muss gelten:

$\theta \leq y_j / |d_j|$ für negative und positive d_j

=> gleichzeitig für alle $j \in B'$ wählbar

definiere x^1 und x^2 aus S durch

$$x_j^1 := \begin{cases} y_j - \theta d_j & j \in B' \\ 0 & \text{sonst} \end{cases} \qquad x_j^2 := \begin{cases} y_j + \theta d_j & j \in B' \\ 0 & \text{sonst} \end{cases}$$

=> $x^1, x^2 \in S$, verschieden von y und

$$y = \frac{1}{2}x^1 + \frac{1}{2}x^2$$

$g : S \to P$ definiert in (3.9) ist linear und $g(y) = y'$

$$\Rightarrow y' = g(y) = g\left(\frac{1}{2}x^1 + \frac{1}{2}x^2\right) = \frac{1}{2}g(x^1) + \frac{1}{2}g(x^2)$$

mit $g(x^1)$, $g(x^2) \in P$ => Widerspruch zu (2)

=> $|B'| \leq m$ => (Annahme 3.1) B' kann zu Basis B erweitert werden

=> y ist zulässige Basislösung zur Basis B

(3) => (1)

sei y zulässige Basislösung von $Ax = b, x \geq 0$ zur Basis B

Lemma 3.5 => es gibt Kostenvektor c, so dass y einzige Optimallösung des LP mit Kostenvektor c ist.

Anders ausgedrückt:

y ist einzige Lösung von (3.12)

$c^T x \leq c^T y =: b_0$,

$Ax = b, \; x \geq 0$

Bringe (3.12) auf Standardform mit Schlupfvariablen x_{n+1} für $c^T x \leq b_0$:

$$\begin{pmatrix} c_1 & \dots & c_n & 1 \\ & & & 0 \\ A_1 & \dots & A_n & \vdots \\ & & & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_{n+1} \end{pmatrix} = \begin{pmatrix} b_0 \\ b \end{pmatrix}$$

=> $B \cup \{n+1\}$ ist Basis von (3.12)

Transformiere (3.12) gemäß der Transformation g aus (3.9)

dies ergibt Ungleichungssystem, dessen m untere Ungleichungen gerade das Polytop P beschreiben.

Die erste Ungleichung wird zu $c_1' x_1' + \dots + c_{n-m}' x_{n-m}' \leq b_0'$ (3.13)

Transformation ist eine Bijektion

=> $y \to y'$ und y' ist einziger Punkt in P, der (3.13) mit Gleichheit erfüllt

=> $c_1' x_1' + \dots + c_{n-m}' x_{n-m}' = b_0'$ ist Stützhyperebene H von P und $H \cap P = \{y'\}$

=> y' ist Ecke von P ❑

Für Kanten gilt entsprechende Charakterisierung (später)

Korollar (Zulässige Lösungen sind Konvexkombination von zulässigen Basislösungen)

Unter den Annahmen 3.1 - 3.3 ist jede zulässige Lösung in S Konvexkombination von zulässigen Basislösungen.

Beweis:

Der zulässige Bereich S kann nach Satz 3.7 als beschränkt angenommen

=> das zugehörige Polyeder P ist ein Polytop

Sei $x \in S$ und x' der zugehörige Punkt in P

=> (Satz von Minkowski) x' ist Konvexkombination von Ecken x'^i von P

=> x ist Konvexkombination der zu den Ecken gehörenden zulässigen BL

denn:

Sei $x' = \sum \lambda_i x'^i$ => $z := \sum \lambda_i x^i \in S$, da S konvex

g linear => $g(z) = \sum \lambda_i g(x^i) = \sum \lambda_i x'^i = x'$

g injektiv, $g(x) = x' = g(z)$ => $x = z$ ❑

Genauere Analyse der Korrespondenz "zulässige Basislösung <-> Ecke"

Ecken x', y' verschieden <=> zugehörige zulässige Basislösungen x, y verschieden

Aber: i.A. nicht die zugehörigen Basen!

d.h. Basen verschieden $\not\Rightarrow$ zugehörige BL verschieden

Beispiel 3.8 (Fortsetzung)

$$A = \begin{pmatrix} 1 & 1 & 1 & \vdots & 1 & & \\ 1 & & & \vdots & & 1 & \\ & & 1 & \vdots & & & 1 \\ & 3 & 1 & \vdots & & & \end{pmatrix} \qquad b = \begin{pmatrix} 4 \\ 2 \\ 3 \\ 6 \end{pmatrix}$$

$B = \{1, 2, 3, 6\}$ $B' = \{1, 2, 4, 6\}$

=> $B^{-1}b = (B')^{-1}b' = (2, 2, 0, 3)^T$. Die zugehörige BL ist in beiden Fällen $(2, 2, 0, 0, 0, 3, 0)^T$

geometrisch:

bzgl. B sind x_4, x_5, x_7 Nichtbasisvariable => $x_4 = x_5 = x_7 = 0$

=> $x' \in H_1 \cap H_2 \cap H_4$

bzgl. B' sind x_3, x_5, x_7 Nichtbasisvariable => $x_3 = x_5 = x_7 = 0$

=> $x' \in H_5 \cap H_2 \cap H_4$

d.h. x' liegt auf mehr als 3 Facetten

in beiden Fällen ist x' dieselbe Ecke $(2,2,0)^T$ und $x = (2, 2, 0, 0, 0, 3, 0)^T$ dieselbe zulässige Basislösung

degenerierte zulässige Basislösung, degenerierte Ecke

= zulässige Basislösung (zu Ecke) mit mehr als $n-m$ Nullen

3.11 Satz (Charakterisierung degenerierter zulässiger BL bzw. Ecken)

x ist zulässige Basislösung zu verschiedenen Basen => x ist degeneriert

x' ist degenerierte Ecke <=> x' ist im Schnitt von mehr als $n-m$ Facetten

Beweis

- Seien B und B' die Basen zu x
  => $x_j = 0$ für die n-m Indices $j \in B$ und für die n-m Indices $j \in B'$
- Übung ❑

Der Hauptsatz der Linearen Optimierung

Sind jetzt so weit um zu zeigen, dass das Optimum eines LP immer (auch) in einer Ecke bzw. zulässigen Basislösung angenommen wird (unter den Annahmen 3.1 - 3.3).

3.12 Satz (Hauptsatz der Linearen Optimierung)

Jede Instanz von LP nimmt das Optimum in einer zulässigen Basislösung an.

Jede Konvexkombination optimaler zulässiger Basislösungen ist optimal.

Beweis

Erste Aussage: geometrisch mit zugehöriger transformierter Zielfunktion $c^Tx \to d^Tx'$

Transformation der Zielfunktion in die geometrische Sicht

Sei S_I gegeben durch $\{ Ax = b, x \geq 0 \}$ und B eine Basis zu zulässiger Basislösung

Betrachte zugehöriges Polytop P zu Basis B gemäß Transformation (3.9)

=> P ist über Nichtbasisvariablen x_N definiert und $x_B = B^{-1}b - B^{-1}A_N x_N$

Für die Zielfunktion c^Tx ergibt sich bei der Transformation:

$$\begin{aligned} c^Tx &= c_B^Tx_B + c_N^Tx_N \\ &= c_B^T(B^{-1}b - B^{-1}A_Nx_N) + c_N^Tx_N \\ &= \underbrace{c_B^TB^{-1}b}_{\text{konstant}} + \underbrace{(c_N^T - c_B^TB^{-1}A_N)}_{d^T} x_N \end{aligned}$$

=> min c^Tx entspricht min d^Tx_N

Beweis aus geometischer Sicht

P abgeschlossen und beschränkt, $y \to d^Ty$ ist stetig auf P

=> Optimum wird auf P angenommen

Sei y^0 optimal und seien y^k, $k = 1, \ldots, r$ die Ecken von P

Satz von Minkowski => $y^0 = \sum \lambda_k y^k$ mit $\lambda_k \geq 0$, $\sum \lambda_k = 1$

Sei j der Index mit d^Ty^j minimal unter den Ecken y^k, $k = 1, \ldots, r$

=> $d^Ty^0 = d^T \sum \lambda_k y^k = \sum \lambda_k d^Ty^k \geq \sum \lambda_k d^Ty^j = d^Ty^j \sum \lambda_k = d^Ty^j$

=> y^j ist optimal => Optimum wird auf einer Ecke angenommen

Korrespondenz Ecken <-> zulässige Basislösungen => Optimum wird auf einer zulässigen Basislösung angenommen

- Die zweite Aussage folgt sofort ❑

- Korollar
  - Jede Optimallösung ist Konvexkombination von optimalen zulässigen Basislösungen
  - Beweis:
    - Sei x eine Optimallösung von $\min c^T x,\ Ax = b, x \geq 0$
    - Korollar zu Satz 3.10 => x ist Konvexkombination von zulässigen BL

      etwa $x = \lambda_1 x^1 + \ldots + \lambda_k x^k$, x^i zulässige BL, $\lambda_i \geq 0$, $\sum \lambda_i = 1$, k endlich
    - => $c^T x = \sum \lambda_i c^T x^i$ (*)
    - Annahme x^r nicht optimal => $c^T x < c^T x^r$

      (*), $\lambda_i \geq 0$ => es gibt in der Konvexkombination eine BL x^s mit $c^T x > c^T x^s$

      => Widerspruch zu x optimal
    - => alle in der Konvexkombination beteiligten BL sind optimal ❑

- Satz 3.12 ist die Grundlage für den Simplexalgorithmus
  - geschickte lokale Suche unter den Ecken (geometrisch)

- geschickte lokale Suche unter den zulässigen Basislösungen (algebraisch)

Hauptpunkte dieses Abschnittes

- Betrachten die Pivotoperation, die die Nachbarschaft von zulässigen Basislösungen definiert
- Analysieren den dahinterstehenden algebraischen Kalkül
- Nehmen dazu Tableaus zur Hilfe in Abschnitt 3.5.

Erste Überlegungen zur Pivotoperation

- Sei y zulässige Basislösung von $Ax = 0,\ x \geq 0$ zur Basis B
  - => können $Ay = b$ schreiben als $\sum_{i \in B} A_{B(i)} y_{B(i)} = b$ (3.14)
- B Basis => jede Spalte $A_j \notin B$ ist als Linearkombinarion der Basisspalten darstellbar
  - => es gibt Zahlen x_{ij} $(i = 1, \ldots, m)$ mit $\sum_{i=1,\ldots,m} A_{B(i)} x_{ij} = A_j$ (3.15)
- (3.14) - $\theta \cdot$(3.15) => $\sum_{i=1,\ldots,m} A_{B(i)} (y_{B(i)} - \theta x_{ij}) + \theta A_j = b$ (3.16)
- Wollen jetzt θ verändern in (3.16) und wieder zu zulässiger Basislösung kommen.
  - Betrachten 3 Fälle:
    - Fall 1: y ist nicht degeneriert und nicht alle $x_{ij} \leq 0$ $(i = 1, \ldots, m)$
      - y nicht degeneriert => alle $y_{B(i)} > 0$ $(i = 1, \ldots, m)$
        Übergang von $\theta = 0$ zu $\theta > 0$ entspricht Übergang von y zu $x(\theta)$ mit

$$x_\ell(\theta) = \begin{cases} y_{B(i)} - \theta x_{ij} & \ell = B(i),\ i = 1, \ldots, m \\ \theta & \ell = j \\ 0 & \text{sonst} \end{cases}$$

=> $x(\theta)$ hat $m+1$ Komponenten > 0 (für kleines θ)

- $x(\theta)$ bleibt zulässige Lösung solange $x(\theta) \geq 0$
  - denn: $Ax(\theta) = \sum_{i=1,\ldots,m} A_{B(i)} (y_{B(i)} - \theta x_{ij}) + \theta A_j = b$ wegen (3.16)
- $x(\theta) \geq 0$ ist erfüllt solange $0 \leq \theta \leq \theta_0$ mit

$$\theta_0 = \min \left\{ \frac{y_{B(i)}}{x_{ij}} \mid x_{ij} > 0,\ i = 1, \ldots, m \right\} \quad (3.17)$$

- Wird das Minimum bei k angenommen, so ist $x(\theta_0)$ eine zulässige Basislösung zur Basis $B' = B - \{B(k)\} \cup \{j\}$
  - denn
    1. die Spalten A_j mit $j \in B'$ sind linear unabhängig
       Annahme nicht. Dann existieren Zahlen d_i mit $\sum_{i=1,\ldots,m,\ i \neq k} d_i A_{B(i)} + d_j A_j = 0$
       (3.15) => $\sum_{i=1,\ldots,m,\ i \neq k} d_i A_{B(i)} + d_j (\sum_{i=1,\ldots,m} A_{B(i)} x_{ij}) = 0$
       dies ist Linearkombination der Spalten aus B zu 0, also müssen alle Koeffizienten 0 sein
       => $(d_i + d_j x_{ij}) = 0$ für $i = 1, \ldots, m,\ i \neq k$ und $d_j x_{kj} = 0$

x_{kj} bestimmt θ_0 => $x_{kj} > 0$ => $d_j = 0$ => $d_i = 0$ für alle i, Widerspruch

2. $x(\theta_0)_{N'} = 0$ nach Konstruktion

3.3 Beispiel (Fortsetzung)

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 4 \\ 1 & & & 1 & 2 \\ & & 1 & & 3 \\ 3 & & 1 & & 6 \end{array}\right) \Rightarrow y_B = \begin{pmatrix} y_3 \\ y_1 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 1 \\ 4 \end{pmatrix}$$

$$A_5 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = A_1 \cdot x_{15} + A_3 \cdot x_{25} + A_6 \cdot x_{35} + A_7 \cdot x_{45}$$

$= 1 \cdot A_1 - 1 \cdot A_3 + 1 \cdot A_6 + 1 \cdot A_7$

=> (3.16) wird zu $(2 - \theta) \cdot A_1 + (2 + \theta) \cdot A_3 + (1 - \theta) \cdot A_6 + (4 - \theta) \cdot A_7 + \theta \cdot A_5$

=> Übergang von $y = x(0)$ zu $x(\theta)$ wird zu

$$y = \begin{pmatrix} 2 \\ 0 \\ 2 \\ 0 \\ 0 \\ 1 \\ 4 \end{pmatrix} \rightarrow x(\theta) = \begin{pmatrix} 2 - \theta \\ 0 \\ 2 + \theta \\ 0 \\ \theta \\ 1 - \theta \\ 4 - \theta \end{pmatrix} \rightarrow \begin{pmatrix} 1 \\ 0 \\ 3 \\ 0 \\ 1 \\ 0 \\ 3 \end{pmatrix} \quad \text{für } \theta = 1$$

x(1) ist zulässige Basislösung zur Basis $B = \{ 3, 1, 5, 7 \}$

Fall 2: y ist degenieriert

dann kann es sein, dass $y_{B(i)} = 0$ und $x_{ij} > 0$ für ein i

=> $\theta_0 = 0$

=> keine Bewegung im R^n und damit im R^{n-m}

=> man bleibt in gleicher Ecke und zulässiger Basislösung, erhält aber eine andere Basis, $A_{B(i)}$ und A_j werden ausgetauscht

Fall 3: alle $x_{ij} \leq 0$ $(i = 1, \ldots, m)$

=> θ kann beliebig groß werden und man bleibt zulässig

=> S_I ist unbeschränkt

=> Zielfunktion ist nach unten unbeschränkt wegen Satz 3.7

Basiswechsel

3.13 Satz (Basiswechsel)

Das Minimum bei der Bildung von θ_0 werde bei $i = k$ angenommen. Dann ist $x(\theta_0)$ eine zulässige Basislösung zur Basis B' mit

$$B'(i) = \begin{cases} B(i) & i \neq k \\ j & i = k \end{cases}$$

Ist k nicht eindeutig, so ist $x(\theta_0)$ degeneriert.

Beweis: folgt aus den vorangestellten Überlegungen

Im Beispiel ist $k = 3$, $B(3) = 6$, $B'(3) = 5$ ❑

Dieser Übergang von zulässiger Basislösung zu zulässiger Basislösung wird Pivotisieren genannt.

Man sagt: $A_{B(k)}$ verlässt die Basis und A_j kommt in die Basis

$x_{B(k)}$ verlässt die Basis und x_j kommt in die Basis

Die Größe θ_0 wird primale Schrittlänge genannt

Zwei zulässige Basislösungen mit unterschiedlicher Basis heißen benachbart, wenn die eine durch eine

Pivotoperation in die andere überführt werden kann. Das Pivotisieren definiert also eine Nachbarschaft auf der Menge der zulässigen Basislösungen.

Ziel: effizienter Übergang von zulässiger Basislösung zu zulässiger Basislösung

=> müssen die x_{ij} verfügbar machen

=> müssen Nichtbasisspalten durch Basisspalten ausdrücken

dies entspricht der Transformation der Basis auf Einheitsmatrix

Ein Beispiel

Ax = b sei gegeben durch

$$3x_1 + 2x_2 + x_3 = 1$$
$$5x_1 + x_2 + x_3 + x_4 = 3$$
$$2x_1 + 5x_2 + x_3 + x_5 = 4$$

Tableau dazu

	x_1	x_2	x_3	x_4	x_5
1	3	2	1	0	0
3	5	1	1	1	0
4	2	5	1	0	1

mit b als 0-ter Spalte

Transformieren Geichungssystem/Tableau bzgl. Basis B so, dass die Basisspalten die Einheitsmatrix bilden.

Für B = { 3, 4, 5 } ergibt dies folgendes Tableau

	x_1	x_2	x_3	x_4	x_5
1	3	2	1	0	0
2	2	-1	0	1	0
3	-1	3	0	0	1

=> $x_{B(1)} = x_3 = 1$

$x_{B(2)} = x_4 = 2$

$x_{B(3)} = x_5 = 3$

Ferner $A_1 = 3A_{B(1)} + 2A_{B(2)} - 1A_{B(3)}$ entspricht Informationen in der Spalte zu x_1

=> die Zahlen x_{ij} stehen im transformierten Tableau in der Spalte j

Falls A_1 in die Basis soll, so ergibt sich

$$\Theta_0 = \min\{\frac{y_{B(i)}}{x_{i1}} \mid x_{i1} > 0,\ i = 1,\dots,m\} = \min\{\frac{1}{3}, \frac{2}{2}\} = \frac{1}{3} \text{ und } k = 1$$

Das Tableau zur neuen Basis entsteht dann durch Transformation so, dass A_1 neuer Einheitsvektor in der Basis wird

	x_1	x_2	x_3	x_4	x_5
$\frac{1}{3}$	1	$\frac{2}{3}$	$\frac{1}{3}$	0	0
$\frac{4}{3}$	0	$-\frac{7}{3}$	$-\frac{2}{3}$	1	0
$\frac{10}{3}$	0	$\frac{11}{3}$	$\frac{1}{3}$	0	1

=> $x_{B'(1)} = x_1 = 1/3$

$x_{B'(2)} = x_4 = 4/3$

$x_{B'(3)} = x_5 = 10/3$

a_{11} war bei diesem Basiswechsel das Pivotelement (wie im Gauss-Algorithmus)

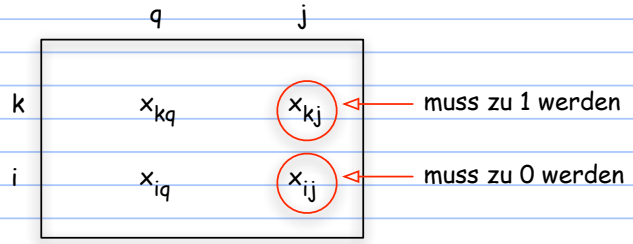
Die Transformationsregeln

Sei (x_{ij}) das Tableau zur Basis B und (x'_{ij}) das Tableau zur neuen Basis B', jeweils inklusive der rechten Seite als 0-te Spalte (x_{i0}) bzw. (x'_{i0}). Sei x_{kj} das Pivotelement. Dann ergeben sich die Einträge im neuen

Tableau gemäß (3.18)

$$\begin{cases} x'_{kq} = \frac{x_{kq}}{x_{kj}} & q = 0,\dots,n \\ x'_{iq} = x_{iq} - x'_{kq}x_{ij} & i = 1,\dots,m,\ i \neq k;\ q = 0,\dots,n \\ B'(i) = B(i) & i = 1,\dots,m,\ i \neq k \\ B'(i) = j & i = k \end{cases}$$

Merkregel



- Ziel: Untersuchung der Veränderung der Zielfunktion beim Basiswechsel
- **Erste Überlegungen zur Veränderung der Zielfunktion**
  - Sei y zulässige Basislösung zur Basis B
    => y hat Zielfunktionswert (Kosten) $z_y = \sum_{i=1,\dots,m} c_{B(i)} y_{B(i)}$
  - Falls A_j in die Basis kommt, so ist $Ax(\theta) = \sum_{i=1,\dots,m} A_{B(i)} (y_{B(i)} - \theta x_{ij}) + \theta A_j \; (= b)$
    $\theta = 1$ => für jede "Einheit" der Variablen x_j, die in die Basis kommt, "verlassen" x_{ij} Einheiten von $y_{B(i)}$ die Basis
    => die Kosten ändern sich um $1 \cdot c_j - \sum_{i=1,\dots,m} x_{ij} c_{B(i)}$
  - Setze $z_j := \sum_{i=1}^{m} x_{ij} c_{B(i)}$ und $\bar{c}_j := c_j - z_j$
    $\bar{c}_j$ heißt reduzierter (oder relativer) Kostenkoeffizient zur Spalte j
    er beschreibt die Kostendifferenz bei Aufnahme von x_j in die Basis mit Wert $x_j = 1$
    z_j beschreibt die Kostenänderung bei den Basisvariablen bei Aufnahme von x_j in die Basis mit Wert $x_j = 1$
  - Wir werden zeigen:
    - $\bar{c}_j < 0$
      => Verbesserung der Zielfunktion erreichbar durch Aufnahme von x_j in die Basis

- $\bar{c}_j \geq 0$ für alle Nichtbasisvariablen j
  => sind in lokalem Optimum bzgl. Basistausch
  => haben globale optimale Basislösung, d.h. die Nachbarschaft basierend auf Basistausch ist exakt

- **Notation**
  - X = momentanes Tableau zur Basis B
    A = Ausgangsmatrix, b anfängliche rechte Seite
    Dann gilt
    - $X = B^{-1}(b|A)$
      d.h. momentanes Tableau entsteht aus $(b|A)$ durch Multiplkation von links mit B^{-1}
    - $z^T = c_B^T X = c_B^T B^{-1} A$
      d.h. diese Formel ergibt die Kostenänderung z_j bei den Basisvariablen
      - denn

$$z_j = \sum_{i=1}^{m} x_{ij} c_{B(i)} = (c_{B(1)}, \ldots, c_{B(m)}) \begin{pmatrix} x_{1j} \\ \vdots \\ x_{mj} \end{pmatrix}$$

$$\Rightarrow z^T = c_B^T X = c_B^T B^{-1} A$$

Optimalitätskriterium

3.14 Satz (Optimalitätskriterium)

Sei x zulässige Basislösung zur Basis B. Dann gilt

(1) Ein Pivotschritt, bei dem x_j mit dem Wert θ_0 in die Basis kommt, ändert die Kosten um den Betrag

$$\theta_0 \bar{c}_j = \theta_0 (c_j - z_j) \qquad (3.19)$$

(2) Gilt

$$\bar{c} = c - z \geq 0 \qquad (3.20)$$

d.h. sind alle reduzierten Kostenkoeffizienten nichtnegativ, so ist x optimal.

Beweis

zu (1) (anschaulich klar nach Vorüberlegungen, hier genaue Rechnung)

Aufnahme von x_j in die Basis ergibt nach den Transformationsregeln (3.18) das folgende neue rechte Seite (= 0te Spalte des neuen Tableaus)

$$x'_{i0} = \begin{cases} x_{i0} - \theta_0 x_{ij} & i \neq k \\ \theta_0 & i = k \end{cases} \qquad \text{mit } \theta_0 = \frac{x_{k0}}{x_{kj}}$$

Sei z der alte Zielfunktionswert und z' der neue

=> neuer Zielfunktionswert ist

$$z' = \sum_{i=1,\, i\neq k}^{m} (x_{i0} - \theta_0 x_{ij}) c_{B(i)} + \theta_0 c_j$$

$$= \underbrace{\sum_{i=1}^{m} x_{i0} c_{B(i)}}_{=z} - \theta_0 \sum_{i=1}^{m} x_{ij} c_{B(i)} - (x_{k0} - \theta_0 x_{kj}) c_{B(k)} + \theta_0 c_j$$

$$= z + \theta_0 \Big(x_{kj} c_{B(k)} + c_j - \underbrace{\sum_{i=1}^{m} x_{ij} c_{B(i)}}_{=c_j - z_j} \Big) - x_{k0} c_{B(k)}$$

$= z + \theta_0(c_j - z_j) + \underbrace{\theta_0 x_{kj} c_{B(k)}}_{= x_{k0} c_{B(k)} \text{ da } \theta_0 = x_{k0}/x_{kj}} - x_{k0} c_{B(k)}$

$= z + \theta_0(c_j - z_j)$

=> (3.19)

zu (2)

Sei y eine zulässige Lösung des LP, d.h. $Ay = b,\ y \geq 0$

$\bar{c} = c - z \geq 0 \Rightarrow c \geq z$. Zusammen mit $y \geq 0$ folgt

$$c^T y \geq z^T y = c_B^T B^{-1} A y = c_B^T B^{-1} b = c_B^T x_B$$

Also ist x global optimal (und insbesondere auch eine beste zulässige Basislösung) ❑

Tableau mit reduzierten Kostenkoeffizienten

Wegen des Optimalitätskriteriums ist es zweckmäßig, die reduzierten Kostenkoeffizienten im Tableau als 0-te Zeile mitzuführen

Frage: Wie bekommen wir sie aus c?

Schreiben Kosten als

$0 = -z_0 + c_1 x_1 + \ldots + c_n x_n$

Dies entspricht einer Erweiterung des Gleichungssystem $Ax = b$ zu

$$\begin{pmatrix} 1 & \vdots & c^T \\ \cdots & \cdots & \cdots \\ 0 & \vdots & A \end{pmatrix} \begin{pmatrix} -z_0 \\ \cdots \\ x \end{pmatrix} = \begin{pmatrix} 0 \\ \cdots \\ b \end{pmatrix}$$

Für Basisvariable x_j folgt dann

$$\bar{c}_j = c_j - z_j = c_j - \sum_{i=0}^{m} x_{ij} c_{B(i)} = 0$$

denn (x_{ij}) ist Einheitsvektor mit 1 an Stelle i_0 mit $B(i_0) = j$

$$\Rightarrow c_j - \sum_{i=0}^{m} x_{ij} c_{B(i)} = c_j - c_{B(i_0)} = c_j - c_j = 0$$

Also $\bar{c}_j = 0$ für Basisvariable

Für Nichtbasisvariable x_j folgt (3.21)

$$\bar{c}_j = c_j - z_j = c_j - \sum_{i=1}^{m} x_{ij} c_{B(i)} = c_j - c_B^T B^{-1} A_j = c_j - c_B^T X_j$$

d.h. $\bar{c}_j$ kann für Nichtbasisvariable aus c und X ermittelt werden

Für $-z_0$ folgt (3.22)

$$-z_0 = -\sum_{i=1}^{m} c_{B(i)} x_{B(i)} = -c_B^T X_0$$

d.h. $-z_0$ kann ebenfalls aus c und X ermittelt werden

Neues, vergrößertes Tableau (im Folgenden immer als Tableau bezeichnet)

	$-z_0$	$\bar{c}_1$	...	$\bar{c}_j$	...	$\bar{c}_n$
	x_{10}	x_{11}	...	x_{1j}	...	x_{1n}
x_B	⋮	⋮		⋮		⋮
	x_{m0}	x_{m1}	...	x_{mj}	...	x_{mn}
	$B^{-1}b$			$B^{-1}A_j$		

3.15 Satz (Aktualisierung der reduzierten Kosten)

Für die Aktualisierung der reduzierten Kostenkoeffizienten bei Basiswechsel gelten dieselben Regeln wie für die anderen Zeilen von X, d.h. analog zu (3.18). Anders ausgedrückt:

Für die in die Basis kommende Spalte j wird auch $x_{0j} = \bar{c}_j$ auf 0 transformiert. Die dadurch verursachten

Änderungen in Zeile 0 ergeben die neuen reduzierten Kostenkoeffizienten und das neue $-z_0$

Beweis: nachrechnen ❑

3.16 Korollar (Test auf unbeschränkte Zielfunktion)

$\bar{c}_j < 0$, alle $x_{ij} \leq 0$ (i = 1,...m) => $c^T x$ ist nach unten unbeschränkt

Beweis

x(θ) kann dann beliebig groß gemacht werden

=> die Zielfunktion fällt um $\theta \cdot \bar{c}_j \to \infty$ ❑

Die Grundversion des Simplexalgorithmus

Algorithmus (Generischer Simplexalgorithmus)

Input

Tableau X eines LP in Standardform mit vollem Zeilenrang und mit Basis als Einheitsmatrix

Output

bei Terminierung: Optimale Basislösung oder Mitteilung, dass die Zielfunktion unbeschränkt ist

Terminierung ist nicht garantiert

- Methode (iteratives Pivotisieren in Richtung geringerer Kosten)
  - while es gibt Spalte j mit $\bar{c}_j < 0$ do
    - wähle Spalte j mit $\bar{c}_j < 0$
    - if $x_{ij} \leq 0$ für alle i then return "Zielfunktion ist unbeschränkt"
    - berechne θ_0 und den Index k, für den das Minimum angenommen wird gemäß (3.17)
    - Pivotisiere mit Pivotelement x_{kj} gemäß (3.18) (Zeile 0 mit gleichen Regeln)
  - return x_B

- Beispiel
  - min $x_1 + x_2 + x_3 + x_4 + x_5$

    unter $3x_1 + 2x_2 + x_3 = 1$

    $5x_1 + x_2 + x_3 + x_4 = 3$

    $2x_1 + 5x_2 + x_3 + x_5 = 4$

    $x_j \geq 0$
  - Ausgangstableau (noch nicht mit reduzierten Kostenkoeffizienten und Basis als Einheitsmatrix)

	x_1	x_2	x_3	x_4	x_5
0	1	1	1	1	1
1	3	2	1	0	0
3	5	1	1	1	0
4	2	5	1	0	1

- Transformieren bzgl. Basis $B = \{ 3, 4, 5 \}$

  d.h. Spalten 3, 4, 5 auf Einheitsvektor bringen inklusive Zeile für Kostenvektor

	x_1	x_2	x_3	x_4	x_5
$-z$	−6	−3	−3	0	0	0
x_3	1	3	2	1	0	0
x_4	2	2	−1	0	1	0
x_5	3	−1	3	0	0	1

- mögliche Art der Transformation:

  1. Zeile 1 von Zeile 2 und von Zeile 3 subtrahieren ergibt Einheitsmatrix in Zeilen 1-3;
  2. dann Zeilen 1-3 von Zeile 0 subtrahieren, dies entspricht genau den Formeln (3.21) und (3.22)

Pivotoperation:

Spalten 1 und 2 haben reduzierte Kostenkoeffizienten < 0, also z.B. x_2 in die Basis bringen.

Berechnung von θ_0 ergibt $\theta_0 = \min \{ 1/2, 3/3 \} = 1/2$ mit $k = 1$. Also verläßt $x_{B(1)} = x_3$ die Basis.

		x_1	x_2	x_3	x_4	x_5
$-z$	$-9/2$	$3/2$	0	$3/2$	0	0
x_2	$1/2$	$3/2$	1	$1/2$	0	0
x_4	$5/2$	$7/2$	0	$1/2$	1	0
x_5	$3/2$	$-11/2$	0	$-3/2$	0	1

Alle reduzierten Kostenkoeffizienten ≥ 0 => Optimallösung erreicht

$x = (0, 1/2, 0, 5/2, 3/2)$ ist optimale Basislösung mit Kosten $z = 9/2$

Applets zum Pivotisieren

Advanced Simplex Pivoting Tool

@ Advanced Simplex Pivot Tool

Matrix Row Operation Tool

@ http://people.hofstra.edu/faculty/Stefan_Waner/RealWorld/tutorialsf1/scriptpivot2.html

Hauptpunkte dieses Abschnitts

- Strategien für die Wahl einer Spalte, die in die Basis soll, falls mehrere möglich sind
- Strategien für die Wahl eines Pivotelements in der Spalte, falls mehrere möglich sind
- Speziell: Erreichen, dass der Simplexalgorithms immer terminiert

Strategien für die Wahl einer Spalte (Pricing)

- Eine Wahl ist nötig, wenn mehrere $\bar{c}_j < 0$ sind, die Wahl beeinflusst die Anzahl der Pivotoperationen.
- Eine eindeutig beste Wahl konnte bisher theoretisch nicht nachgewiesen werden, kommerzielle Solver verwenden eine ganze Reihe heuristischer Kriterien, darunter folgende:
  - (1) Größter relativer Abstieg unter den Nichtbasisvariablen
    - Wähle unter allen Spalten mit $\bar{c}_j < 0$ ein j mit maximalem $|\bar{c}_j|$
    - Diese Spalte hat den größten relativen Abstieg der Kostenfunktion in Richtung einer Nichtbasisvariablen. Da θ_0 nicht bekannt ist, garantiert dies nicht den größten Abstieg.
  - (2) Größter absoluter Abstieg unter den Nichtbasisvariablen
    - Wähle unter allen Spalten mit $\bar{c}_j < 0$ ein j mit maximalem $|\theta_0 \bar{c}_j|$
    - Diese Spalte hat den größten absoluten Abstieg der Kostenfunktion in Richtung einer Nichtbasisvariablen.

Führt in der Regel zu einer Reduktion der Anzahl der Pivotoperationen, hat aber einen um den Faktor m höheren Aufwand gegenüber (1)

- (3) Größter relativer Abstieg im gesamten zulässigen Bereich
  - Wähle unter allen Spalten mit $\bar{c}_j < 0$ ein j mit maximalem

$$\frac{\bar{c}_j}{\sqrt{1+\sum_{i=1}^{m} x_{ij}^2}} = \frac{\bar{c}_j}{\|x(\theta)-x\|} \text{ mit } \theta = 1$$

  - Diese Spalte hat den größten relativen Abstieg der Kostenfunktion in Richtung $x(\theta) - x$

Strategien für die Wahl des Pivotelements

- Falls $\theta_0 > 0$, so spielt die Pivotwahl (im Sinne der Endlichkeit des Verfahrens) keine große Rolle, da eine neue Ecke/Basislösung mit besserem Zielfunktionswert erreicht wird. Wichtig ist sie, falls $\theta_0 = 0$, da dann die momentane Basislösung degeneriert ist und die Pivotoperation zwar die Basis ändert, aber nicht die Basislösung/ Ecke, vgl. Satz 3.13
- Dadurch kann Kreiseln entstehen, d.h. ausgehend von Tableau X erreicht man nach mehreren Pivotoperationen wieder Tableau X

  => man hat einen "Zykel" verschiedener Basen zu derselben Basislösung durchlaufen

=> der Simplexalgorithmus terminiert nicht

3.15 Beispiel (Ein Beispiel für Kreiseln, Gass 1964)

min $-3/4\, x_1 + 150\, x_2 - 1/50\, x_3 + 6\, x_4$

unter $1/4\, x_1 - 60\, x_2 - 1/25\, x_3 + 9\, x_4 \le 0$

$1/2\, x_1 - 90\, x_2 - 1/50\, x_3 + 3\, x_4 \le 0$

$x_3 \le 1$

$x_j \ge 0$

Ausgangstableau mit Basis aus Schlupfvariablen

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	0	−3/4	150	−1/50	6	0	0	0
x_5	0	1/4	−60	−1/25	9	1	0	0
x_6	0	1/2	−90	−1/50	3	0	1	0
x_7	1	0	0	1	0	0	0	1

Wählt man nacheinander die Pivotelemente

$x_{11}\ x_{22}\ x_{13}\ x_{24}\ x_{15}\ x_{26}$

so gelangt man wieder zum Ausgangstableau

Beweis: nachrechnen, z.B. mit dem Matrix Row Operation Tool

@ http://people.hofstra.edu/faculty/Stefan_Waner/RealWorld/tutorialsf1/scriptpivot2.html

Regeln zur Vermeidung von Kreiseln

Die lexikographische Auswahlregel

Teile jede Zeile i, für die $x_{ij} > 0$ durch x_{ij} und wähle als Pivotzeile unter diesen die lexikographisch kleinste

Lexikographische Regel:

$$\text{Wähle k so, dass } \frac{1}{x_{kj}} \cdot X_{k,\cdot} = \text{lex-min} \left\{ \frac{1}{x_{ij}} \cdot X_{i,\cdot} \mid i = 1, \ldots, m,\ x_{ij} > 0 \right\}$$

mit $X_{i,\cdot}$ = i-te Teile von Tableau X und lex-min = lexikographisches Minimum

$$\theta_0 := \frac{x_{k0}}{x_{kj}}$$

3.16 Satz (Lexikographische Auswahlregel vermeidet Kreiseln)

(a) Die Zeilen $i = 1, \ldots, m$ können im Tableau lexikographisch positiv gemacht werden

(b) Sind die Zeilen i = 1, ..., m im Tableau lexikographisch positiv, so terminiert der Simplexalgorithmus mit lexikographischer Auswahlregel nach endlich vielen Schritten.

Beweis

(1) Lexikographisch positive Zeilen können erreicht werden

z.B. durch Permutation der Basispalten an die Positionen 1,2,...,m

=> jede Zeile beginnt entweder mit dem Eintrag $x_{i0} > 0$ oder mit (0,...,0,1,...), wobei $x_{i,B(i)} = 1$

=> jede Zeile ist lexikographisch positiv, d.h. $>_{lex}$ (0,0,...,0)

(2) Das lex-min ist eindeutig bestimmt

Annahme nicht => es gibt Zeilen i, r mit

$$\frac{1}{x_{ij}}(x_{i0}, x_{i1}, \dots, x_{in}) = \frac{1}{x_{rj}}(x_{r0}, x_{r1}, \dots, x_{rn})$$

=> die Zeilen i und r sind linear abhängig

=> Widerspruch zu Annahme 3.1 dass rang(A) = m

(3) Alle Zeilen des Tableaus bleiben nach jedem Pivotschritt lexikographisch positiv

Ist x_{kj} das Pivotelement, so ergeben sich als neue Zeilen

für i = k

$$\frac{1}{x_{kj}}(x_{k0}, x_{k1}, \dots, x_{kn})$$

$x_{kj} > 0$ => man bleibt lexikographisch positiv

für $i \neq k$

$$(x_{i0}, x_{i1}, \dots, x_{in}) - x_{ij}\frac{1}{x_{kj}}(x_{k0}, x_{k1}, \dots, x_{kn})$$

Diese Zeile ist $>_{lex}$ (0,0,...,0)

$$\Leftrightarrow \frac{1}{x_{kj}}(x_{k0}, x_{k1}, \dots, x_{kn}) <_{lex} \frac{1}{x_{ij}}(x_{i0}, x_{i1}, \dots, x_{in})$$

Dies ist der Fall, da die Zeile k das lex-min ist und das lex-min eindeutig ist (Gleichheit kann nicht auftreten)

(4) Die 0-te Zeile (Kostenzeile) nimmt bei jedem Pivotschritt streng lexikographisch zu

Ist x_{kj} das Pivotelement, so ergibt sich die neue 0-te Zeile durch die Pivotoperation als

$$(-z, \bar{c}_1, \ldots, \bar{c}_n) - \frac{1}{x_{kj}} \bar{c}_j (x_{k0}, x_{k1}, \ldots, x_{kn})$$

> 0, da Pivotelement — < 0, da reduzierte Kosten der Pivotspalte — lex positiv

=> neue Zeile ist lexikographisch größer

(5) Terminierung

Eine Basis bestimmt die 0-te Zeile eindeutig wegen (3.21) und (3.22).

0-te Zeile wächst lexikographisch => stets verschiedene Basislösungen

es gibt nur endlich viele Basislösungen => Terminierung ❑

3.15 Beispiel (Fortsetzung)

Das Ausgangstableau ist bereits lexikographisch positiv

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	0	-3/4	150	-1/50	6	0	0	0
x_5	0	1/4	-60	-1/25	9	1	0	0
x_6	0	1/2	-90	-1/50	3	0	1	0
x_7	1	0	0	1	0	0	0	1

Die Folge

$$x_{11}\ x_{22}\ x_{23}\ x_{34}\ x_{35}$$

erfüllt die lexikographische Auswahlregel und terminiert mit dem Optimum. Nach der Pivotoperation mit x_{34} wird die degenerierte Basislösung verlassen.

Beweis: nachrechnen, z.B. mit dem Matrix Row Operation Tool

@ http://people.hofstra.edu/faculty/Stefan_Waner/RealWorld/tutorialsf1/scriptpivot2.html

Die Regel von Bland

Wähle unter allen Spalten j mit $\bar{c}_j < 0$ die mit kleinstem Spaltenindex j

Wähle unter allen Zeilen i mit $\theta_0 = x_{i0}/x_{ij}$ die mit kleinstem Spaltenindex $B(i)$

Beweis: Übung ❑

- Empirische Tests zeigen, dass die Regel von Bland sehr viele Pivotoperationen benötigt

- Vermeidung von Kreiseln in der Praxis
  - Die praktische Erfahrung zeigt, dass Kreiseln selten auftritt und zum Teil bereits durch das notwendige Runden behoben wird. Kommerzielle Solver kontrollieren den Fortschritt in der Zielfunktion und gehen bei Bedarf (d.h. Verdacht auf Kreiseln) zu einer Pivotstrategie über, die das Kreiseln vermeidet, bis eine andere Basislösung erreicht wird.

- Ziel: Dem Simplexalgorithmus ein Start-Tableau zu geben, also ein Tableau, das bzgl. einer zulässigen Basislösung …
  - Dies ist einfach wenn das LP in der Form $Ax \le b,\ x \ge 0$ mit $b \ge 0$ gegeben ist. Dann braucht man nur Schlupfvariable $s_1, \ldots, s_m$ einzuführen und die zugehörigen Spalten bilden eine Basis

Start-Tableau

		x_1	…	x_n	s_1	s_2	…	s_m
$-z$	0	c_1	…	c_n	0	0	…	0
s_1	b_1	a_{11}	…	a_{1n}	1	0	…	0
⋮	⋮	⋮	⋱	⋮	⋮	⋮	⋱	0
s_m	b_m	a_{m1}	…	a_{mn}	0	0	…	1

- Geschieht für allgemeine LP in Standardform mit der Zwei-Phasen-Methode
  - Gegeben: LP in Standardform

$$\min\ c^T x$$
$$\text{unter}\quad Ax = b$$
$$x \ge 0$$

  und o.B.d.A $b \ge 0$ (sonst Zeilen mit -1 multiplizieren)
  - Phase I

Führe künstliche Variable $x^a = (x_1^a, ..., x_m^a)^T$ ein (^{a} für artificial) und löse das LP

$$\begin{aligned} \min\ \xi &= x_1^a + \ldots + x_m^a \\ \text{unter}\quad & x^a + Ax = b \qquad (3.23) \\ & x^a, x \geq 0 \end{aligned}$$

mit dem Simplexalgorithmus (die künstlichen Variablen bilden eine Basis und die Kostenkoeffizienten bzgl. ξ sind in reduzierter Form (3.21))

3 mögliche Ergebnisse

Fall 1: Das Minimum von ξ ist 0 und alle x_i^a sind Nichtbasisvariablen

=> alle $x_i^a = 0$ und wir haben eine zulässige Basislösung für das Ausgangsproblem

Fall 2: Das Minimum von ξ ist 0 aber nicht alle x_i^a sind Nichtbasisvariablen

Sei $x_{B(i)}$ eine künstliche Variable.

$\xi = 0$ => $x_{B(i)} = 0$

Versuche, $x_{B(i)}$ aus der Basis zu entfernen.

Dazu muss im Tableau X eine nicht-künstliche Nichtbasisspalte j mit $x_{ij} \neq 0$ existieren

(da $x_{B(i)} = 0$ können wir dann eine Pivotoperation durchführen, bei $x_{ij} < 0$ vorher Zeile mit -1 multiplizieren)

Fall 2a: Es existiert eine nicht-künstliche Nichtbasisspalte j mit $x_{ij} \neq 0$

Pivotisiere mit dem Pivotelement x_{ij}

=> haben eine Basis mit weniger künstlichen Variablen

Fall 2b: Es existiert keine nicht-künstliche Nichtbasisspalte j mit $x_{ij} \neq 0$

=> $x_{ij} = 0$ für $j = 1, ..., n$ (d.h. für alle "echten" Spalten j)

(klar für Nichtbasisvariable x_j, Basisvariable x_j haben in der zugehörigen Spalte an Position i eine 0, da ja $x_{B(i)}$ dort die 1 hat)

=> $\text{rang}(A) < m$

=> Annahme 3.1 nicht erfüllt

Insbesondere ist dann die i-te Zeile von A eine Linearkombination der anderen Zeilen von A und kann inklusive der künstlichen Variablen $x_{B(i)}$ gestrichen werden

damit können wir Annahme 3.1 fallen lassen, Phase I gibt uns den Test auf $\text{rang}(A) = m$

Wiederholung dieser Schritte liefert am Ende eine zulässige Basislösung des Ausgangs LP (möglicherweise nach Streichung einiger linear abhängiger Zeilen aus A, die verbleibenden Zeilen haben vollen Rang)

Fall 3: Das Minimum von ξ ist > 0

=> es gibt für das gegebene LP keine zulässige Lösung, d.h. $S = \emptyset$

- damit können wir Annahme 3.2 fallen lassen, Phase I gibt uns den Test auf $S \neq \emptyset$
- Phase II
  - macht weiter mit der zulässigen Basislösung aus Phase I (falls $S \neq \emptyset$)
  - muss noch die reduzierten Kostenkoeffizienten $\bar{c}_j$ und den Zielfunktionswert $-z_0$ bzgl. der ursprünglichen Kostenfunktion ermitteln
    - entweder mit (3.21) und (3.22)
    - oder durch Mittransformation der Kostenkoeffizienten c_j in Phase I

      Satz 3.15 => sie stehen am Beginn von Phase II bereits in reduzierter Form zur Verfügung

- Algorithmus (Zwei-Phasen-Methode)
  - Input
    - LP in Standardform
  - Output
    - bei Terminierung: Optimale Basislösung oder Mitteilung, dass die Zielfunktion unbeschränkt ist oder $S = \emptyset$ ist
    - Terminierung kann durch Auswahlregeln garantiert werden

- Methode
  - Phase I
    - füge künstliche Variable $x_1^a, \ldots, x_m^a$ ein
    - call Simplexalgorithmus mit Kostenfunktion $\xi = \sum x_i^a$
    - if $\xi_{opt} > 0$ then return "es gibt keine zulässige Lösung"
    - if Basis enthält künstliche Variable
      - then
        - if diese künstliche Variable nicht herauspivotisiert werden kann
          - then
            - streiche zugehörige Zeile
            - call Phase I für das resultierende LP
      - else // $\xi_{opt} = 0$ und die Basis enthält keine künstlichen Variablen
        - call Phase II
  - Phase II
    - call Simplexalgorithmus mit der ursprünglichen Kostenfunktion bzgl. der Basis aus Phase I

3.17 Satz (Zwei-Phasen-Methode)

Die Zwei-Phasen-Methode löst jedes LP in Standardform. Auf die Annahmen (3.1)-(3.3) kann dabei verzichtet werden.

Beweis

klar nach den Vorüberlegungen ❑

Beispiel

min $x_1 + x_2 + x_3 + x_4 + x_5$

unter $3x_1 + 2x_2 + x_3 = 1$

$5x_1 + x_2 + x_3 + x_4 = 3$

$2x_1 + 5x_2 + x_3 + x_5 = 4$

$x_j \geq 0$

Ausgangstableau für Phase I, eigentliche Zielfunktion wird mittransformiert

	x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z$	0	0	0	0	1	1	1	1	1
$-\xi$	0	1	1	1	0	0	0	0	0
x_1^a	1	1	0	0	3	2	1	0	0
x_2^a	3	0	1	0	5	1	1	1	0
x_3^a	4	0	0	1	2	5	1	0	1

Für den Start der Phase I müssen die reduzierten Kostenkoeffizienten der künstlichen Variablen bzgl. ξ auf 0 gebracht werden (reduzierte Kosten sind 0 für Basisvariable).

Wird erreicht durch Subtraktion der letzten 3 Zeilen von der ξ-Zeile (z-Zeile hat bereits reduzierte Kosten 0 in den Basisspalten)

Starttableau für Phase I

	x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z$	0	0	0	0	1	1	1	1	1
$-\xi$	-8	0	0	0	-10	-8	-3	-1	-1
x_1^a	1	1	0	0	3	2	1	0	0
x_2^a	3	0	1	0	5	1	1	1	0
x_3^a	4	0	0	1	2	5	1	0	1

- Pivotisieren mit den jeweils rot eingekreisten Pivotelementen ergibt

	x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z$	-1/3	-1/3	0	0	0	1/3	2/3	1	1
$-\xi$	-14/3	-10/3	0	0	0	-4/3	1/3	-1	-1
x_1	1/3	1/3	0	0	1	2/3	1/3	0	0
x_2^a	4/3	-5/3	1	0	0	-7/3	-2/3	1	0
x_3^a	10/3	-2/3	0	1	0	11/3	1/3	0	1

	x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z$	-1/2	-1/2	0	0	-1/2	0	1/2	1	1
$-\xi$	-4	4	0	0	2	0	1	-1	-1
x_2	1/2	1/2	0	0	3/2	1	1/2	0	0
x_2^a	5/2	-1/2	1	0	7/2	0	1/2	1	0
x_3^a	3/2	-15/6	0	1	-11/2	0	-3/2	0	1

	x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z$	-3	0	-1	0	-4	0	0	0	1
$-\xi$	-3/2	7/2	1	0	11/2	0	3/2	0	-1
x_2	1/2	1/2	0	0	3/2	1	1/2	0	0
x_4	5/2	-1/2	1	0	7/2	0	1/2	1	0
x_3^a	3/2	-5/2	0	1	-11/2	0	-3/2	0	1

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
-z	-9/2	5/2	-1	-1	3/2	0	3/2	0	0
-ξ	0	1	1	1	0	0	0	0	0
x_2	1/2	1/2	0	0	3/2	1	1/2	0	0
x_4	5/2	-1/2	1	0	7/2	0	1/2	1	0
x_5	3/2	-5/2	0	1	-11/2	0	-3/2	0	1

Wir haben den schönen Effekt, dass das Tableau am Ende von Phase I bereits optimal für Phase II ist (alle reduzierten Kostenkoeffizienten der eigentlichen Zielfunktion sind ≥ 0)

Ziel: Den Simplexalgorithmus geometrisch interpretieren

wird sich als Übergang von Ecke zu Ecke herausstellen, mit gelegentlichen mehrfachen Pivotoperationen in degenerierter Ecke

3.8 Beispiel (Fortsetzung)

max $x_1 + 14x_2 + 6x_3$ <=> min $- x_1 - 14x_2 - 6x_3$

unter $x_1 + x_2 + x_3 \leq 4$

$x_1 \leq 2$

$x_3 \leq 3$

$3x_2 + x_3 \leq 6$

$x_j \geq 0$

Der Simplexalgorithmus erzeugt nach Einführung von Schlupfvariablen (= erste Basis) die nachstehende Folge von Tableaus:

1

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	0	-1	-14	-6	0	0	0	0
x_4	4	1	1	1	1	0	0	0
x_5	2	1	0	0	0	1	0	0
x_6	3	0	0	1	0	0	1	0
x_7	6	0	3	1	0	0	0	1

2

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	2	0	-14	-6	0	1	0	0
x_4	2	0	1	1	1	-1	0	0
x_1	2	1	0	0	0	1	0	0
x_6	3	0	0	1	0	0	1	0
x_7	6	0	3	1	0	0	0	1

3

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	14	0	-8	0	6	-5	0	0
x_3	2	0	1	1	1	-1	0	0
x_1	2	1	0	0	0	1	0	0
x_6	1	0	-1	0	-1	1	1	0
x_7	4	0	2	0	-1	1	0	1

4

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	30	0	0	8	14	-13	0	0
x_2	2	0	1	1	1	-1	0	0
x_1	2	1	0	0	0	1	0	0
x_6	3	0	0	1	0	0	1	0
x_7	0	0	0	-2	-3	3	0	1

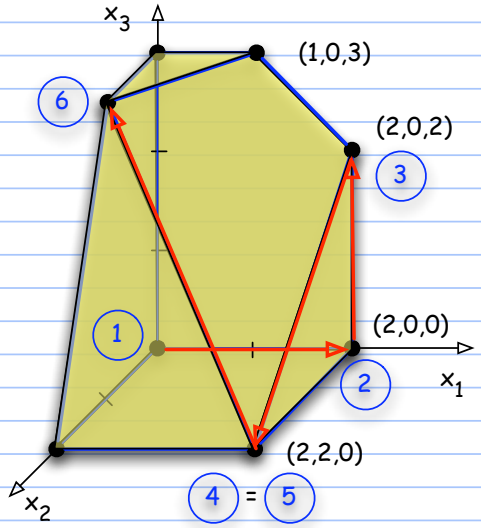
5

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	30	0	0	-2/3	1	0	0	13/3
x_2	2	0	1	1/3	0	0	0	1/3
x_1	2	1	0	2/3	1	0	0	-1/3
x_6	3	0	0	1	0	0	1	0
x_5	0	0	0	-2/3	-1	1	0	1/3

6

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-z$	32	1	0	0	2	0	0	4
x_2	1	-1/2	1	0	-1/2	0	0	1/2
x_3	3	3/2	0	1	3/2	0	0	-1/2
x_6	0	-3/2	0	0	-3/2	0	1	1/2
x_5	2	1	0	0	0	1	0	0

Dies entspricht folgender Folge von Ecken im zugehörigen Polytop in den Variablen x_1, x_2, x_3



Wir werden dieses Verhalten jetzt genauer untersuchen

Adjazenz von Ecken und zulässigen Basislösungen

Zwei Ecken x', y' eines Polyeders P heißen adjazent

:<=> die Strecke $[x', y']$ ist Kante des Polyeders P

Zwei zulässige Basislösungen x, y von $Ax = b, x \geq 0$ heißen adjazent

:<=> wenn man mit einer Pivotoperation gemäß Satz 3.16 von B_x zu B_y kommt

Beachte:

(1) dann kommt man auch von B_y zu B_x, d.h. die Nachbarschaft ist symmetrisch

(2) es gibt dann Spalten j und k mit $B_y = (B_x - \{A_j\}) \cup \{A_k\}$

(3) dies lässt zu, dass $x = y$ ist, die Basislösung also degeneriert ist

Nach dieser Definition durchläuft der Simplexalgorithmus eine Folge adjazenter zulässiger Basislösungen $x^1, x^2, \ldots, x^N$ mit

$c^T x^1 \geq c^T x^2 \geq \ldots \geq c^T x^N = z_{opt}$

3.18 Satz (Interpretation von Kanten in den drei Sichten)

Sei P ein Polytop und $S = \{x \mid Ax = b, x \geq 0\}$ die zugehörige zulässige Menge eines LP in Standardform.

Seien $x' = (x_1', \ldots, x_{n-m}')^T$, $y' = (y_1', \ldots, y_{n-m}')^T \in P$ verschiedene Ecken und seien $x = (x_1, \ldots, x_n)^T$, $y = (y_1, \ldots, y_n)^T \in S$ die zugehörigen zulässigen Basislösungen gemäß (3.7).

Dann sind folgende Aussagen äquivalent:

(1) $[x', y']$ ist Kante von P

(2) Ist $z' \in [x', y']$ strikte Konvexkombination von Punkten u', v' aus P, so sind $u', v' \in [x', y']$

(3) x, y sind (verschiedene) adjazente zulässige Basislösungen von S

Beweis

ähnlich zu Satz 3.10, nutzt die Transformationen $P \leftrightarrow S$

(1) => (2)

Sei $[x', y']$ Kante von P => es gibt eine Stützhyperebene H mit $H \cap P = [x', y']$, etwa $H = \{ w' \mid h^T w' = g \}$.

Annahme, (2) gilt nicht, sei dann o.B.d.A. $u' \notin [x', y']$ und $h^T u' < g$.

Dann ist $h^T v' \leq g$, da ganz P in einem Halbraum von H liegt

=> $g = h^T z' = h^T(\lambda u' + (1-\lambda)v') = \lambda h^T u' + (1-\lambda)h^T v' < g$ da $\lambda \neq 0$, Widerspruch

(2) => (3)

Annahme, x', y' erfüllen (2), aber nicht (3).

Seien M_x, M_y die Mengen der Spalten A_j von A mit $x_j > 0$ bzw. $y_j > 0$.

Claim: es gibt eine zulässige Basislösung $w \neq x, y$ mit $w_j > 0$ => $A_j \in M_x \cup M_y$

Annahme nicht. Sei o.B.d.A. $y \neq 0$ (möglich wegen $x \neq y$). Wähle dann

$$c_j := \begin{cases} 0 & A_j \in M_y \\ 1 & A_j \in M_x - M_y \\ nM & \text{sonst} \end{cases}$$

wobei M so groß gewählt ist, dass $c_j u_j > n$ für jede zulässige Basislösung u (ein solches M existiert wegen Lemma 3.4).

=> y ist einzige Optimallösung zu c und jede zulässige Basislösung u mit Komponenten $u_j > 0$ nicht in $M_x \cup M_y$ hat größere Kosten als x

=> Simplex gestartet in x würde (da x und y nicht adjazent sind) bzgl. c feststellen, dass keine Verbesserung möglich ist und x als optimal ausweisen, Widerspruch

Der zu w gehörige Punkt $w' \in P$ ist eine Ecke und liegt daher nicht in $[x', y']$

=> w liegt nicht in $[x, y]$

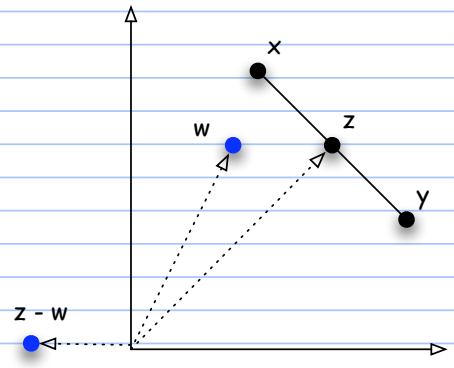
Sei $z := 1/2 (x + y)$ => $z_j > 0$ für alle $A_j \in M_x \cup M_y$ => jede Komponente kann etwas verkleinert werden.

Sei $d := z - w$ => $d_j \neq 0$ nur für Komponenten in $M_x \cup M_y$

$z_j > 0$ für alle Komponenten in $M_x \cup M_y$ => es gibt $\theta > 0$ mit $u := z + \theta d$, $v := z - \theta d \geq 0$

Nachrechnen ergibt $Au = b$ und $Av = b$, d.h. $u, v \in S$.

Ferner: w nicht in $[x, y]$ => u, v nicht in $[x, y]$



=> z = 1/2 (u + v) ist Konvexkombination der zulässigen Punkte u, v ∉ [x, y]

=> (Transformation nach P) z´ = 1/2 (u´ + v´) ist Konvexkombination der Punkte u´, v´ ∉ [x´, y´],

Widerspruch

(3) => (1)

Seien B_x, B_y die Basen zu x und y mit $B_y = (B_x - \{ A_j \}) \cup \{ A_k \}$

Sei c definiert durch

$$c_j := \begin{cases} 0 & \text{falls } A_j \in B_x \cup B_y \\ 1 & \text{sonst} \end{cases}$$

Claim: x und y sind die einzigen optimalen zulässigen Basislösungen sind bzgl c

klar: x, y sind optimal bzgl. c

Annahme es gibt weitere optimale zulässige Basislösung z

Konstruktion von c => z erfüllt $z_j > 0$ => $A_j \in B_x \cup B_y$

=> $B_z \subseteq B_x \cup B_y$

$A_j \notin B_x$ => (mit $| B_x \cup B_y | = m + 1$, $z \neq x$) $A_j \in B_z$

=> $B_z = (B_x - \{ A_q \}) \cup \{ A_j \}$

=> Man kann von zulässiger Basislösung x durch Pivotisieren sowohl zu y (A_j rein, A_k raus) als auch zu z (A_j rein, A_q raus) gelangen. In beiden Fällen kommt A_j rein.

Sei $B_x(r) = k$ und $B_x(s) = q$

$$\Rightarrow \quad \theta_0 = \frac{x_{r0}}{x_{rj}} = \frac{x_{s0}}{x_{sj}}$$

=> y = z, Widerspruch

Claim => nur Konvexkombinationen von x, y erfüllen

$Aw = b$

$w \geq 0$

$c^T w \leq c^T x$

Transformation auf P wie im Beweis zu Satz 3.10 =>

in P erfüllen nur Punkte w´ aus [x´, y´] die Ungleichung $d^T w' \leq d^T x'$ (d´ = transformierter Kostenvektor)

=> [x´, y´] ist Durchschnitt eines Halbraum mit P

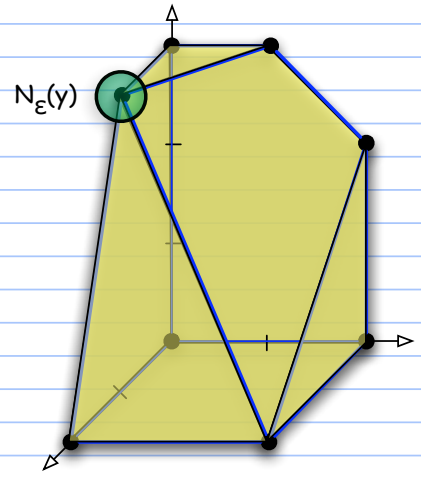
=> [x´, y´] ist Kante von P ❑

3.19 Bemerkungen

(1) LP ist ein konvexes Optimierungsproblem. Also ist wegen Satz 2.16 die Euklidische Nachbarschaft

$$N_\varepsilon(y) := \{ x \in S_I : || y-x || \leq \varepsilon \}$$

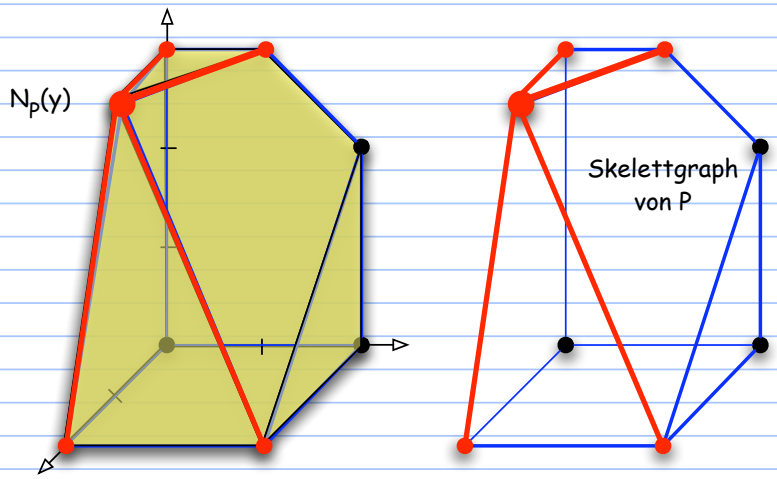
exakt.



(2) Die geometrische Interpretation des Simplexalgorithmus definiert eine andere Nachbarschaft, nämlich über Adjazenz von Ecken:

$$N_p(y) := \{ x \mid x \text{ Ecke von } P,\ x \text{ adjazent zu } y \}$$

Diese ist ebenfalls exakt wegen Satz 3.18, Satz 3.10 und Satz 3.14 und entspricht genau der Graphennachbarschaft im Skelettgraphen des Polytops.

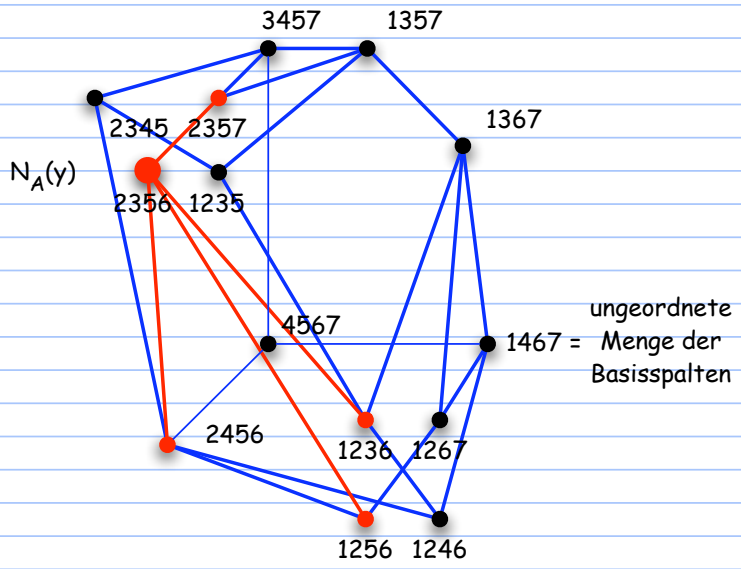


(3) Die algebraische Interpretation des Simplexalgorithmus definiert eine dritte Nachbarschaft, nämlich über Adjazenz von zulässigen Basislösungen:

$N_A(y) := \{ x \mid x$ zulässige Basislösung, x adjazent zu $y \}$

Diese ist ebenfalls exakt im Sinne von Satz 3.14 (keine negativen reduzierten Kosten) und entspricht ebenfalls einer Graphennachbarschaft, und zwar in dem Graphen, der aus dem Skelettgraphen durch Verfeinerung von Knoten (= Ecke) zu einer Menge von Knoten (= alle zulässigen Basislösungen zu der Ecke) ergibt. Er stimmt mit

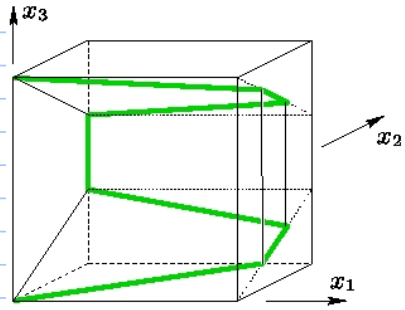
dem Skelettgraphen überein, wenn P keine degenerierten Ecken hat.



(4) Der Simplexalgorithmus kann also als lokale Suche auf dem durch N_A definierten Graphen angesehen werden, die Überprüfung der Nachbarschaft auf Verbesserung ist über die Vorzeichen der reduzierten Kosten möglich, also sehr einfach.

(5) Es ist offen, ob diese lokale Suche polynomial ist. Zu allen einfachen Auswahlregeln gibt es bislang Gegenbeispiele, die exponentiell viele Schritte (Pivotoperationen) erfordern.

Diese Gegenbeispiele sind in der Regel sogenannte Klee-Minty Würfel, d.h. leicht verzerrte würfel, auf denen der Simplexalgorithmus alle Ecken abläuft, obwohl er schon in einem Schritt fertig sein könnte.



@ http://www.mathematik.de/ger/information/forschungsprojekte/zieglergeometrie/zieglergeometrie.html

(6) Diese Gegenbeispiele gehören jedoch nicht zu den praktisch relevanten Problemen. In der Praxis ist die Laufzeit des Simplexalgorithmus linear in der Anzahl der Nebenbedingungen.

4. Dualität 21

Die duale Form eines LP in allgemeiner Form

Herleitung der dualen Form

Betrachte ein LP in allgemeiner Form: (4.1)

$$\min c^T x \qquad x \in \mathbb{R}^n, c \in \mathbb{R}^n$$

$$\begin{array}{lllll} \text{unter} & a_i^T x & = & b_i & i \in M \quad a_i \in \mathbb{R}^n \\ & a_i^T x & \geq & b_i & i \in \overline{M} \\ & x_j & \geq & 0 & j \in N \\ & x_j & & \text{beliebig} & j \in \overline{N} \end{array}$$

Wir transformieren es auf Standardform gemäß Lemma 3.2 mit

Überschussvariablen x_i^s für die Ungleichungen

Aufspaltung $x_j = x_j^+ - x_j^-$ mit $x_j^+, x_j^- \geq 0$

Dies ergibt

$$\left.\begin{array}{l} \min \quad \hat{c}^T \hat{x} \\ \text{unter} \quad \hat{A}\hat{x} = b, \hat{x} \geq 0 \quad \text{mit} \\ \hat{A} = \left(A_j, j \in N \;\middle|\; (A_j, -A_j), j \in \overline{N} \;\middle|\; \begin{array}{l} 0, \quad i \in M \\ -I, \quad i \in \overline{M} \end{array} \right) \\ \hat{x} = \left(x_j, j \in N \mid (x_j^+, x_j^-), j \in \overline{N} \mid x_i^s, i \in \overline{M} \right)^T \\ \hat{c} = \left(c_j, j \in N \mid (c_j, -c_j), j \in \overline{N} \mid 0, i \in \overline{M} \right)^T \end{array}\right\} \quad (4.2)$$

wobei o.B.d.A. die Matrix $\hat{A}$ vollen Zeilenrang habe, und A_j die Koeffizientenspalte zu x_j in (4.1) bezeichnet

Die bisherige Theorie des Simplexalgorithmus ergibt:

Falls eine optimale Lösung von (4.2) existiert, dann existiert auch eine Basis $\hat{B}$ von $\hat{A}$ mit

$$\hat{c}^T - \underbrace{(\hat{c}_{\hat{B}}^T \hat{B}^{-1})}_{=:\pi^T} \hat{A} \geq 0$$

d.h. reduzierten Kosten ≥ 0

$\Rightarrow$ $\pi^T := \hat{c}_{\hat{B}}^T \hat{B}^{-1} \in \mathbb{R}^m$ ist zulässige Lösung bzgl. der Ungleichungen

$$\pi^T \hat{A} \le \hat{c}^T \qquad (4.3)$$

mit m = Anzahl der Restriktionen in (4.1)

Die Ungleichungen (4.3) haben 3 Teile bzgl. der Spalteneinteilung:

1. Teil

$$\pi^T A_j \le c_j, \quad j \in N \qquad (4.4)$$

2. Teil

$$\left.\begin{array}{rcl} \pi^T A_j & \le & c_j \\ -\pi^T A_j & \le & -c_j \end{array}\right\} \Leftrightarrow \pi^T A_j = c_j, \quad j \in \overline{N} \qquad (4.5)$$

3. Teil

$$-\pi_i \le 0 \Leftrightarrow \pi_i \ge 0, \quad i \in \overline{M} \qquad (4.6)$$

Definition des dualen LP

(4.4) - (4.6) definieren Restriktionen für ein neues LP in den Variablen $\pi_1, \ldots, \pi_m$. Ausgestattet mit der Zielfunktion $\max \pi^T b$ heißt dieses das duale LP zum Ausgangsproblem (4.1). Das Ausgangsproblem (4.1) heißt primales LP.

Transformationsregeln (ergeben sich aus (4.4) - (4.6))

Primal		Dual
$\min c^T x$		$\max \pi^T b$
$a_i^T x = b_i$	$i \in M$	π_i beliebig
$a_i^T x \ge b_i$	$i \in \overline{M}$	$\pi_i \ge 0$
$x_j \ge 0$	$j \in N$	$\pi^T A_j \le c_j$
x_j beliebig	$j \in \overline{N}$	$\pi^T A_j = c_j$

Beachte: Das duale LP wurde aus den Optimalitätsbedingungen des primalen gewonnen. Die Variablen $\pi_1, \ldots, \pi_m$ entsprechen also Multiplikatoren der Zeilen von $\hat{A}$ so dass das Optimalitätskriterium erfüllt ist.

4.1 Satz (dual dual = primal)

Das duale Problem des dualen ist das primale.

Daher sprechen wir auch von primal-dualen Paaren von LPs

Beweis

Schreibe das duale in primaler Form:

$$\begin{array}{lcll} \min\ \pi^T(-b) & & \text{unter} & \\ (-A_j^T)\pi & \geq & -c_i & j \in N \\ (-A_j^T)\pi & = & -c_i & j \in \overline{N} \\ \pi_i & \geq & 0 & j \in \overline{M} \\ \pi_i & & \text{beliebig} & j \in M \end{array}$$

Aus den Transformationsregeln ergibt sich als duales LP

$$\begin{array}{lcll} \max\ x^T(-c) & & \text{unter} & \\ x_j & \geq & 0 & j \in N \\ x_j & & \text{beliebig} & j \in \overline{N} \\ -a_i^T x & \leq & -b_i & i \in \overline{M} \\ -a_i^T x & = & -b_i & i \in M \end{array}$$

was offenbar gleich dem primalen LP ist ❑

Der Dualitätssatz

4.2 Satz (Schwacher und Starker Dualitätssatz)

Sei x primal zulässig und π dual zulässig. Dann gilt (schwacher Dualitätssatz)

$$c^T x \geq \pi^T b \qquad (4.7)$$

Hat ein LP eine optimale Lösung, so auch das duale, und die Optimalwerte sind gleich (starker Dualitätssatz)

Beweis

Sei x primal zulässig und π dual zulässig. Dann folgt

$$c^T x \overset{\pi \text{ dual zulässig}}{\geq} (\pi^T A)x = \pi^T(Ax) \overset{x \text{ primal zulässig}}{\geq} \pi^T b$$

O.B.d.A. sei das LP in primaler Form gegeben und habe eine optimale Lösung

=> in der Form (4.2) hat es eine optimale zulässige Basislösung $\hat{x}$ mit zugehöriger Basis $\hat{B}$

$\Rightarrow$ $\pi^T = \hat{c}_{\hat{B}}^T \hat{B}^{-1}$ ist nach Konstruktion dual zulässig

Für dieses π ergibt sich

$$\pi^T b = (\hat{c}_{\hat{B}}^T \hat{B}^{-1})b = \hat{c}_{\hat{B}}^T(\hat{B}^{-1}b) = \hat{c}_{\hat{B}}^T \hat{x}_B = \hat{c}^T \hat{x}$$

Also haben π und $\hat{x}$ denselben Zielfunktionswert.

Aus (4.7) folgt, das π dual optimal ist ❑

4.3 Satz (mögliche primal-duale Paare)

Für jedes primal-duale Paar treten genau folgende Situationen auf

(1) beide LPs haben eine endliche Optimallösung und die Optimalwerte sind gleich

(2) beide LPs haben keine zulässige Lösung

(3) das eine hat eine unbeschränkte Zielfunktion, das andere hat keine zulässige Lösung

primal \ dual	endliche Optimallösung	zulässige Lösung, unbeschränkte Zielfunktion	keine zulässige Lösung
endliche Optimallösung	(1)		
zulässige Lösung, unbeschränkte Zielfunktion			(3)
keine zulässige Lösung		(3)	(2)

Beweis

Starker Dualitätssatz => In Zeile 1 und Spalte 1 der Tabelle tritt Fall (1) auf, und dieser tritt nur an Position (1,1) auf

Betrachte Zeile 2 der Tabelle, d.h. x ist primale Lösung aber $c^T x$ ist nach unten unbeschränkt.

Falls eine dual zulässige Lösung π existiert, so folgt $\pi^T b \leq c^T x$ mit dem schwachen Dualitätssatz

=> $c^T x$ ist nach unten beschränkt => Widerspruch.

Also kann in Zeile 2 der Tabelle nur (3) auftreten, und nur an Position (2,3)

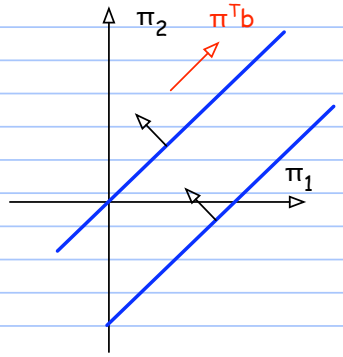
Entsprechend für Spalte 2.

Beispiel für (3)

(P) min x_1 unter $x_1 + x_2 \geq 1$, $-x_1 - x_2 \geq 1$, $x_1, x_2 \geq 0$

=> zweite Ungleichung nicht erfüllbar => (P) hat keine zulässigen Lösungen

(D) max $\pi_1 + \pi_2$ unter $\pi_1 - \pi_2 \leq 1$, $\pi_1 - \pi_2 \leq 0$, $\pi_1, \pi_2 \geq 0$

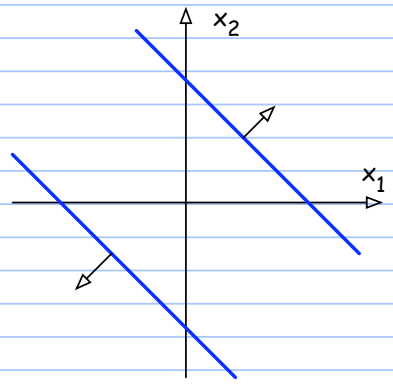


=> $\pi^T b$ ist unbeschränkt

Also bleibt nur Eintrag (3,3) der Tabelle übrig. Dieser Fall kann auftreten

Beispiel für (2)

(P) min x_1 unter $x_1 + x_2 \geq 1$, $-x_1 - x_2 \geq 1$, x_1, x_2 beliebig



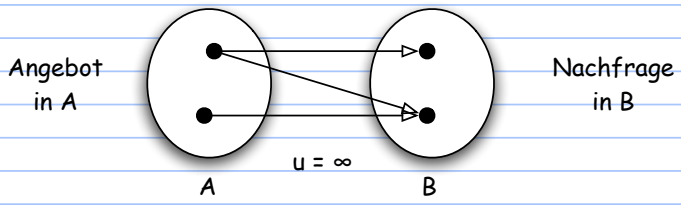
=> (P) hat keine zulässigen Lösungen

(D) max $\pi_1 + \pi_2$ unter $\pi_1 - \pi_2 = 1$, $\pi_1 - \pi_2 = 0$, $\pi_1, \pi_2 \geq 0$

Beide Gleichungen sind nicht erfüllbar => (D) hat keine zulässigen Lösungen ❑

Das Transportproblem und das zugehörige duale Problem

Hitchcock Problem oder Transportproblem (Hitchcock 1941) ist ein speziellen Minimum Cost Flow Problem, vgl. ADM I



G "bipartit"

Transportiere ein Gut (Öl, Getreide, Kohle) kostengünstig von Angebotsorten zu Nachfrageorten.

Knoten $i \in A$ $(i = 1, ..., m)$ bietet a_i Einheiten an

Knoten $j \in B$ $(j = 1, ..., n)$ fragt b_j Einheiten nach, gesamtes Angebot = gesamter Nachfrage.

Kanten $(i,j) \in A \times B$ haben Kosten c_{ij} pro transportierter Einheit und unendliche Kapazität u_{ij}

Eine LP Formulierung für das Transportproblem

x_{ij} = transportierte Einheiten von i nach j

min $\sum_{i,j} c_{ij} x_{ij}$ unter

$\sum_j x_{ij} = a_i$ für alle i (alles wird in Knoten i abtransportiert)

$\sum_i x_{ij} = b_j$ für alle j (alles kommt in Knoten j an)

$x_{ij} \geq 0$ für alle i, j

Die zugehörige Koeffizientenmatrix A hat folgende Form

	x_{11}	x_{12}	…	x_{1n}	x_{21}	x_{22}	…	x_{2n}	…	x_{m1}	x_{m2}	…	x_{mn}
$i = 1,\dots,m$	1	1	…	1	0	0	…	0	…	0	0	…	0
	0	0	…	0	1	1	…	1	…	0	0	…	0
			⋱				⋱		⋱			⋱	
	0	0	…	0	0	0	…	0	…	1	1	…	1
$j = 1,\dots,n$	1	0	…	0	1	0	…	0	…	1	0	…	0
	0	1	…	0	0	1	…	0	…	0	1	…	0
			⋱				⋱		⋱			⋱	
	0	0	…	1	0	0	…	1	…	0	0	…	1

Das zugehörige duale LP

Führe duale Variable u_i, v_j für folgende Form der Nebenbedingungen ein

$u_i \quad -\sum_j x_{ij} = -a_i$ für alle i

$v_j \quad \sum_i x_{ij} = b_j$ für alle j

Dann lautet das duale LP

$\max \sum_i -a_i u_i + \sum_j b_j v_j$ unter

$-u_i + v_j \le c_{ij}$ für alle i, j

u_i, v_j beliebig

Interpretation des dualen LP

"Dualer" Unternehmer bietet an, die Transporte durchzuführen

u_i = Preis zum Auskaufen im Angebotsort pro Einheit

v_j = Erlös am Zielort pro Einheit

$v_j - u_i$ = Gewinn für Transport einer Einheit von i nach j

$v_j - u_i \le c_{ij}$ Unterbieten der primalen Kosten ist notwendig um Auftrag für die Strecke (i,j) vom primalen Unternehmer zu bekommen

Unter diesen Bedingungen will der duale Unternehmer seinen Gewinn $\sum_j b_j v_j - \sum_i a_i u_i$ maximieren

Das duale Problem zum Diätenproblem

Das primale Problem (vgl. Beispiel 3.1)

$\min c^T x$

unter $Ax \ge r$

$x \ge 0$

Das zugehörige duale Problem

$\max \pi^T r$

unter $\pi^T A \leq c^T$

$\pi^T \geq 0$

Interpretation

Pillenfabrikant stellt Pillen für die verschiedenen m Ingredienten her (Magnesium, Vitamin C, ...)

verlangt Preis π_i pro Einheit von Ingredient i

$\pi^T A_j \leq c_j$ <=> Preise aller Pillen, die eine Einheit von Nahrungsmittel j ersetzen, darf den Preis c_j einer Einheit von Nahrungsmittel j nicht überschreiten (sonst keine Marktchancen)

$\max \pi^T r$ <=> Gewinnmaximierung des Pillenfabrikant

In vielen ökonomischen Problemen haben duale LPs eine natürliche Interpretation.

Dies sind einfache notwendige und hinreichende Bedingungen für die Optimalität eines Paares von primal ...

4.4 Satz (Komplementärer Schlupf)

Sei x primal zulässig und π dual zulässig. Dann sind folgende Aussagen äquivalent:

x, π sind optimal (primal bzw. dual)

$u_i := \pi_i \cdot (a_i^T x - b_i) = 0$ für alle $i = 1, ..., m$ (4.8)

$v_j := (c_j - \pi^T A_j) \cdot x_j = 0$ für alle $j = 1, ..., n$ (4.9)

d.h. : (Schlupf primaler/dualer Restriktion)·(Wert zugehöriger dualer/primaler Variable) = 0

Beweis

$u_i \geq 0$

denn

$a_i^T x - b_i = 0 \Rightarrow u_i = 0$

$a_i^T x - b_i \geq 0 \Rightarrow \pi_i \geq 0 \Rightarrow u_i \geq 0$

$v_j \geq 0$

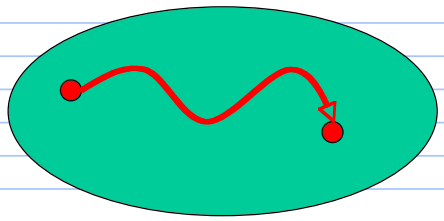
denn

x_j beliebig $\Rightarrow \pi^T A_j = c_j \Rightarrow v_j = 0$

$x_i \geq 0 \Rightarrow \pi^T A_i \leq c_i \Rightarrow v_i \geq 0$

- Setze $u := \sum_i u_i$, $v := \sum_j v_j$ => $u, v \geq 0$. Dann folgt
  - $u = 0$ <=> (4.8) gilt
  - $v = 0$ <=> (4.9) gilt
- Nun ist
  - $u + v = \sum_i \pi_i \cdot (a_i^T x - b_i) + \sum_j (c_j - \pi^T A_j) \cdot x_j$
  - $= - \sum_i \pi_i b_i + \sum_j c_j x_j + \sum_i \pi_i a_i^T x - \sum_j \pi^T A_j x_j$
  - $= - \pi^T b + c^T x + \pi^T (Ax) - (\pi^T A) x$
  - $= - \pi^T b + c^T x$
  - Also: $u + v = - \pi^T b + c^T x$
- Es gelten (4.8) und (4.9) => $u + v = 0$ => $c^T x = \pi^T b$
  - schwacher Dualitätssatz => x, π sind optimal
- x, π seien optimal
  - starker Dualitätssatz => $c^T x = \pi^T b$ => $u + v = 0$ => (4.8) und (4.9) ❑

- **Das kürzeste Wege Problem als primales LP**
  - Shortest Path Problem (SP)
    - Instanz
      - Digraph G
      - reellwertige Kantengewichte $c(e)$, $e \in E(G)$
      - Knoten $s, t \in V(G)$
    - Aufgabe
      - Bestimme elementaren s,t-Weg W mit minimalem Gewicht $c(W)$ (kürzester s,t-Weg)
        - $c(W) = \sum_{e \in E(W)} c(e)$



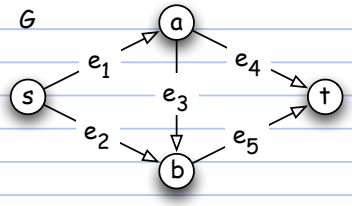
  - (SP) ist eine Instanz von (LP)

Die Knoten-Kanten-Inzidenzmatrix $A = (a_{ij})$ von G ist definiert durch

$$a_{ij} := \begin{cases} +1 & \text{falls } i \xrightarrow{e_j} \\ -1 & \text{falls } \xrightarrow{e_j} i \\ 0 & \text{sonst} \end{cases}$$

wobei $V(G) = \{ 1, \ldots, n \}$ und $E(G) = \{ e_1, \ldots, e_m \}$

Beispiel



A

	e_1	e_2	e_3	e_4	e_5
s	1	1	0	0	0
t	0	0	0	−1	−1
a	−1	0	1	1	0
b	0	−1	−1	0	1

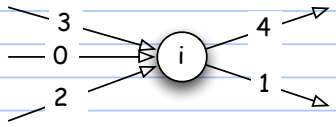
Die Knoten-Kanten-Inzidenzmatrix eines Digraphen hat pro Spalte genau eine 1, genau eine -1 und ansonsten Nullen

=> Summe der Zeilen ist 0 => rang(A) < n

Später: rang(A) = n-1 falls G zusammenhängend (im ungerichteten Sinne) ist

Sei f_j eine Variable, die Fluss durch die Kante e_j repräsentiert, und $f = (f_1, \ldots, f_m)^T$

Dann bedeutet Flusserhaltung im Knoten i, dass $a_i^T f = 0$



Einfluss in i = 5 = Ausfluss aus i

Ein s,t-Weg ist dann ein Fluss der Stärke 1 (alle $f_j = 1$ auf dem Weg und 0 sonst) von s nach t

=> jeder s,t-Weg erfüllt das Gleichungssystem

$$Af = b \text{ mit } b = \begin{pmatrix} +1 \\ -1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \begin{matrix} \text{Zeile s} \\ \text{Zeile t} \\ \\ \\ \\ \end{matrix} \quad \left.\begin{matrix} \\ \\ \\ \\ \\ \end{matrix}\right\} \text{Flusserhaltung}$$

Natürlich gibt es auch Lösungen dieses Gleichungssystem, die keinen s,t-Wegen entsprechen. Es gilt jedoch

4.6 Lemma

(1) Falls

$$\min\ c^T f$$
$$Af = b$$
$$f \geq 0$$

eine optimale Lösung hat, so auch eine mit $f_j \in \{0, 1\}$. Jede solche Lösung entspricht einem s,t-Weg

(2) Der Simplexalgorithmus ermittelt eine solche Lösung

Beweis:

(1) folgt aus den Algorithmen für kostenminimale s,t-Flüsse aus ADM I

(2) folgt aus der Tatsache, dass die Matrix A vollständig unimodular ist, was impliziert, dass alle zulässigen Basislösungen des LP ganzzahlig sind. Dies wird im Kapitel über Ganzzahlige Lineare Optimierung gezeigt ❑

Lösung von (SP) mit dem Simplexalgorithmus

Wir modellieren (SP) als (LP)

$$\min\ c^T f$$

$$Af = b \quad (A = \text{Knoten-Kanten-Inzidenzmatrix})$$
$$f \geq 0$$

und lösen es mit dem Simplexalgorithmus.

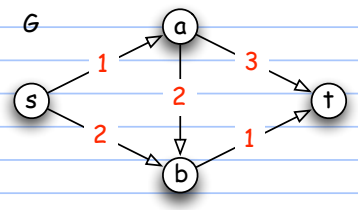
Da $\text{rang}(A) < n$, können wir eine Zeile weglassen

=> lasse Zeile zum Knoten t weg, dann wird $b \geq 0$

Im Beispiel ergeben sich bzgl. des Kostenvektors $c = (1, 2, 2, 3, 1)$ folgende Tableaus

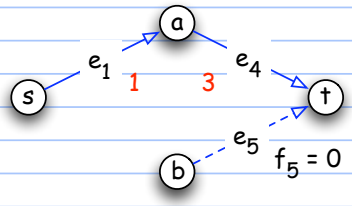
Ausgangstableau, noch nicht auf Basis transformiert und Graph mit Kosten

	f_1	f_2	f_3	f_4	f_5
	1	2	2	3	1
s	1	1	1	0	0	0
a	0	-1	0	1	1	0
b	0	0	-1	-1	0	1



Wähle $\{1, 4, 5\}$ als Basis und transformiere Tableau auf Basisform, stelle Basislösung im Graphen dar

	f_1	f_2	f_3	f_4	f_5
	-4	0	-1	0	0	0
f_1	1	1	1	0	0	0
f_4	1	0	1	1	1	0
f_5	0	0	-1	-1	0	1

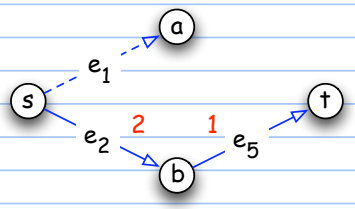


Die Basislösung hat n-1 = |V| - 1 Variablen, aber nicht jeder s,t-Weg hat so viele Kanten

=> viele zulässige Basislösungen sind degeneriert (ein Standard-Phänomen bei Kombinatorischen Optimierungsproblemen)

Nächstes Tableau und Basislösung im Graphen

	f_1	f_2	f_3	f_4	f_5
	-3	0	0	1	1	0
f_1	0	1	0	-1	-1	0
f_2	1	0	1	1	1	0
f_5	1	0	0	0	1	1



=> Optimallösung erreicht, kürzester Weg hat Länge 3

Das duale Problem zum Kürzeste-Wege-Problem

formuliert bzgl. Tableau mit Zeile zum Knoten t

=> Dualvariable π_i entsprechen Knotenpotenzial

Tableau im Beispiel

	f_1	f_2	f_3	f_4	f_5	b
c	1	2	2	3	1	
π_s	1	1	0	0	0	+1
π_t	0	0	0	-1	-1	-1
π_a	-1	0	1	1	0	0
π_b	0	-1	-1	0	1	0

Duales LP:

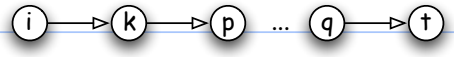
$\max \pi_s - \pi_t$

$\pi_i - \pi_j \leq c_{ij}$ für alle Kanten $(i, j) \in E(G)$

π_i beliebig

Interpretation des dualen LP

Entlang eines Weges



von i nach t gilt

$(\pi_i - \pi_k) + (\pi_k - \pi_p) + \ldots + (\pi_q - \pi_t) = \pi_i - \pi_t$

$\leq c_{ik} \qquad \leq c_{kp} \qquad \leq c_{qt}$

$\Rightarrow \pi_i - \pi_t \leq c_{ik} + c_{kp} + \ldots + c_{qt}$ = Länge des Weges von i nach t

Da dies für jeden Weg gilt, folgt

$\pi_i - \pi_t \leq$ Länge eines kürzesten Weges von i nach t

$\Rightarrow \max \pi_s - \pi_t$ bedeutet

finde möglichst große untere Schranke für die gesuchte kürzeste Weglänge

Bedingungen vom komplementären Schlupf

Weg f und Knotenpotenzial π sind primal-dual optimal $\Leftrightarrow$

(1) $f_{ij} > 0 \Rightarrow \pi_i - \pi_j = c_{ij}$

d.h. Kante (i,j) liegt auf kürzestem Weg => Potenzialdifferenz = Kosten

(2) $\pi_i - \pi_j < c_{ij} \Rightarrow f_{ij} = 0$

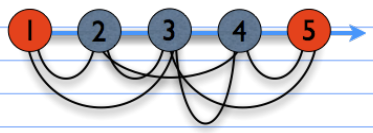
d.h. Potenzialdifferenz < Kosten => Kante (i,j) liegt nicht auf kürzestem Weg

Interpretation:

Entlang eines kürzesten Weges sind die unteren Schranken $\pi_i - \pi_t$ scharf

Bindfaden Interpretation

Kante (i, j) <-> Bindfaden der Länge c_{ij}

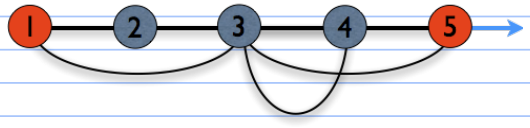


$\pi_i - \pi_j$ <-> Auseinanderziehen der Endknoten

$\pi_i - \pi_j \leq c_{ij}$ <-> Auseinanderziehen ist durch Länge c_{ij} beschränkt

$\max \pi_s - \pi_t$ <-> ziehe s und t soweit wie möglich auseinander

Komplementärer Schlupf: genau die strammen Bindfäden liegen auf kürzestem Weg



- Bemerkungen
  - Streichen der Zeile zu t => keine Variable π_t => duale Zielfunktion ist $\max \pi_s$

    Aber: Kanten (i,t) ergeben die duale Bedingung $\pi_i \leq c_{it}$, so dass (indirekt) π_s nicht beliebig wachsen kann.

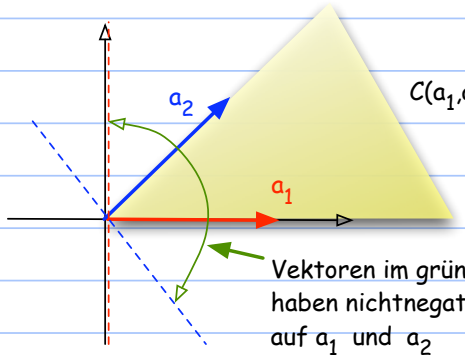
    Dieselbe duale Bedingung $\pi_i \leq c_{it}$ ergibt sich, wenn man $\pi_t = 0$ setzt, was man o.b.d.A. immer tun kann, da im dualen nur Potenzialdifferenzen auftreten.
  - Der Dijkstra Algorithmus (ADM I) angewendet auf den dualen Graphen (alle Kanten umgedreht) berechnet iterativ die π_i, wobei $\pi_t = 0$ gesetzt wird.

- Ist ein zentrales Lemma im Bereich "Dualität", taucht in verschiedenen Varianten auf, die man auch Alternativsätze nennt. Ist ein sehr nützliches Lemma.

- Kegel und Projektionen
  - Der von $a_1, \ldots, a_m$ erzeugte Kegel $C(a_1, \ldots, a_m)$
    - Seien $a_1, \ldots, a_m \in R^n$ (z.B. die Zeilenvektoren von A). Der von $a_1, \ldots, a_m$ erzeugte Kegel $C(a_1, \ldots, a_m)$ ist definiert durch

      $$C(a_1, \ldots, a_m) := \{ x \in \mathbb{R}^n \mid x = \sum_{i=1}^{m} \pi_i a_i,\ \pi_i \geq 0 \}$$

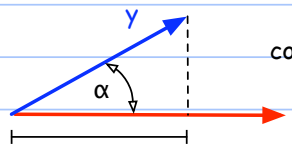
      = Menge der nichtnegativen Linearkombinationen von $a_1, \ldots, a_n$



Vektoren im grünen Winkel
haben nichtnegative Projektionen
auf a_1 und a_2

Die Projektion von y auf a

Projektion von y auf a



$$\cos\alpha = \frac{y^T a}{\|y\| \cdot \|a\|}$$

Projektion von y auf a = $\|y\| \cos\alpha$

=> die Projektion von y auf a ist nichtnegativ <=> $y^T a$ ist nichtnegativ

4.5 Satz (Farkas Lemma)

Seien $a_1, \ldots, a_m \in R^n$ und $c \in R^n$. Dann sind äquivalent

(1) für alle $y \in R^n$ gilt: $y^T a_i \geq 0$ für alle $i = 1,\ldots,m$ => $y^T c \geq 0$

d.h. für alle y gilt:

y hat nichtnegative Projektion auf alle a_i

=> y hat nichtnegative Projektion auf c

(2) $c \in C(a_1, \ldots, a_m)$

d.h. c liegt im von $a_1, \ldots, a_m$ erzeugten Kegel

Beweis

(1) => (2)

Betrachte das LP

$$\min\ c^T y$$
$$a_i^T y \geq 0 \quad i = 1,\ldots,m$$
$$y \text{ beliebig}$$

=> $y = 0$ ist zulässige Lösung des LP

Die Zielfunktion ist wegen (1) nach unten beschränkt, denn die rote Bedingung impliziert $c^T y \geq 0$

=> LP hat endliche Optimallösung

=> das duale LP

max 0

$\pi^T A_j = c_j$

$\pi \geq 0$

hat eine zulässige Lösung

=> es gibt Zahlen $\pi_1, \ldots, \pi_m \geq 0$ mit $c = \pi^T A = \sum_i \pi_i a_i$

=> $c \in C(a_1, \ldots, a_m)$

(2) => (1)

$c \in C(a_1, \ldots, a_m)$ => es gibt Zahlen $\pi_i \geq 0$ mit $c = \sum_i \pi_i a_i$

Betrachte y mit $y^T a_i \geq 0$ für alle $i = 1,\ldots m$

=> $y^T c = \sum_i \pi_i y^T a_i \geq \sum_i \pi_i \cdot 0 = 0$ ❑

Es gibt viele äquivalente Formulierungen zum Farkas Lemma. Beispiele sind

(A) $\forall y \; (y^T a_i \geq 0 \; \forall i \Rightarrow y^T b \geq 0) \Leftrightarrow \exists \; x \geq 0$ mit $A^T x = b$ (Originalversion von Farkas 1894)

(B) $\forall y \geq 0 \; (y^T a_i \geq 0 \; \forall i \Rightarrow y^T b \geq 0) \Leftrightarrow \exists \; x \geq 0$ mit $A^T x \leq b$

Weitere in Kap 7.5

Eine Anwendung des Farkas Lemma: notwendige Bedingungen für das Disjunkte-Wege-Problem

Disjunkte-Wege-Problem

Instanz

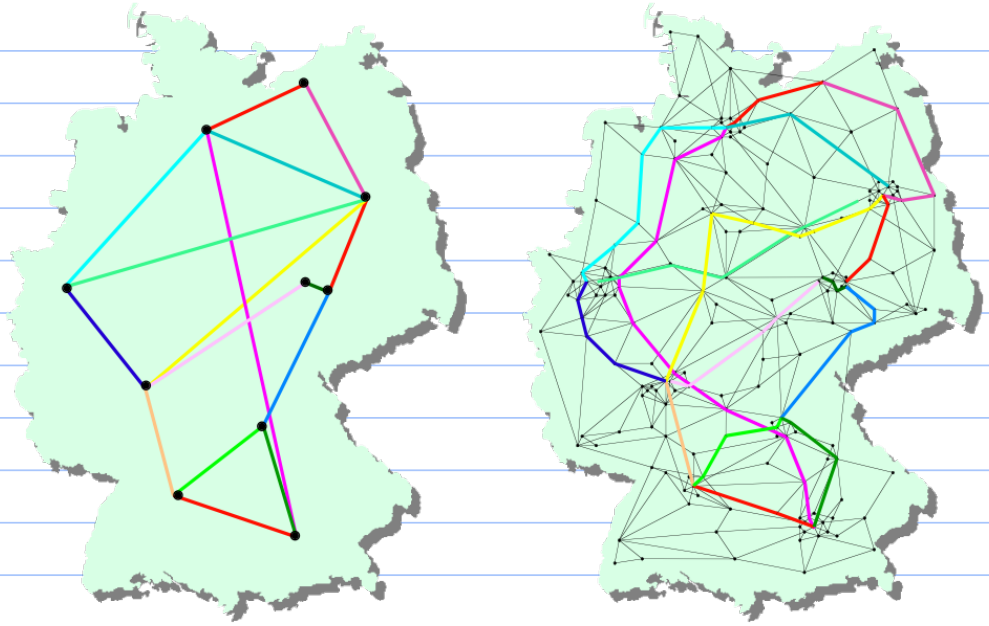
Ungerichteter Graph G

Knotenpaare $\{ s_1, t_1 \}, \ldots, \{ s_k, t_k \}$

Aufgabe

Bestimme paarweise kantendisjunkte Wege von s_i nach t_i $(i = 1, \ldots, k)$

Ein Beispiel: Kostenoptimale Einbettungen von VPNs in das Basisnetz der Telekom



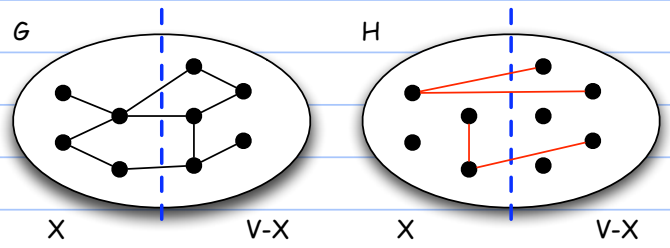
Die Entscheidungsversion des Disjunkte-Wege-Problem ist NP-vollständig. Daher sind möglichst starke notwendige und hinreichende Kriterien für die Existenz einer Lösung gesucht.

Schnittkriterium

Sei H der Graph mit $V(H) := V(G)$ und $E(H) := \{ \{ s_1, t_1 \}, \ldots, \{ s_k, t_k \} \}$. Notwendig für die Existenz einer Lösung ist die Bedingung (Schnittkriterium)

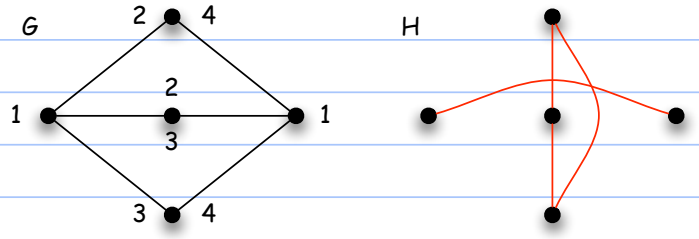
$$|\delta_G(X)| \geq |\delta_H(X)| \text{ fur alle } \emptyset \neq X \subseteq V(G)$$

d.h. aus X müssen in G mindestens so viele Kanten rausgehen, wie Paare aus H verbunden werden müssen



Das Schnittkriterium ist nicht hinreichend

4.6 Beispiel



Schnittkriterium ist erfüllt, aber es gibt keine Lösung

Distanzkriterium

Sei $dist_{G,z}(s,t)$ die Länge eines kürzesten Weges von s nach t in G bzgl. einer Kantenbewertung $z(e) \geq 0$, $e \in E(G)$.

Eine Instanz des Disjunkte-Wege-Problem erfüllt das Distanzkriterium

:<=> für jede Kantenbewertung $z(e) \geq 0$, $e \in E(G)$, gilt

$$\sum_{\{s,t\} \in E(H)} dist_{G,z}(s,t) \leq \sum_{e \in E(G)} z(e)$$

Das Schnittkriterium ergibt sich als Spezialfall des Distanzkriteriums für die Kantenbewertung

$$z(e) := \begin{cases} 1 & \text{falls } e \in \delta(X) \\ 0 & \text{sonst} \end{cases}$$

4.7 Satz (Notwendigkeit des Distanzkriteriums)

Das Distanzkriterium ist notwendig und hinreichend für die Existenz einer fraktionalen Lösung des Disjunkte-Wege-Problem.

Insbesondere ist es notwendig für die Existenz einer Lösung des Disjunkte-Wege-Problem

Beweis

Betrachte das Disjunkte-Wege-Problem als ein Kreis-Packungs-Problem

Kreise = alle elementaren Kreise in $G + H$, die genau eine Kante von H enthalten

k = Anzahl dieser Kreise

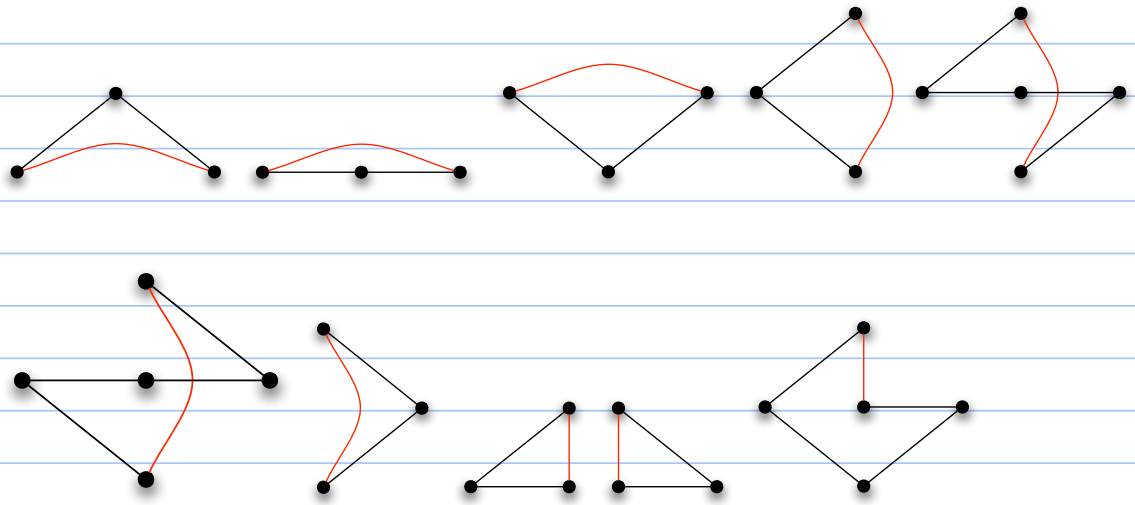
ganzzahlige Kreis-Packung = Vereinigung von paarweise kantendisjunkten Kreisen, die jede Kante aus H genau einmal enthalten

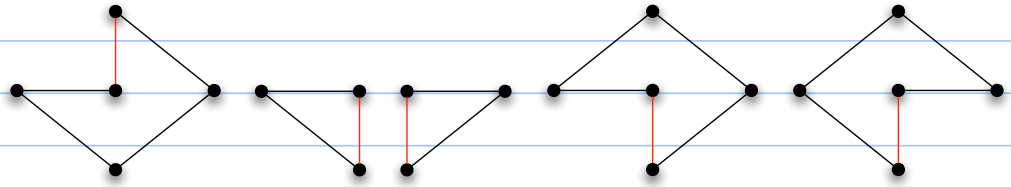
(Existenz <=> Lösbarkeit des Disjunkte-Wege-Problem)

fraktionale Kreis-Packung = nichtnegative Linearkombination (der Inzidenzvektoren) all dieser Kreise, so dass der resultierende Vektor bei den Kanten aus H genau den Wert 1 ergibt und an keiner Kante aus G den Wert 1 überschreitet.

(enthält ganzzahlige Kreis-Packungen als Spezialfall)

In Beispiel 4.6 gibt es die folgenden Kreise im Kreis-Packungsproblem





Formalisierung der fraktionalen Kreis-Packung

Sei M die $E(G)$-Kreis-Inzidenzmatrix, d.h.

Zeilen von M <-> Kanten von G

Spalten von M <-> Inzidenzvektoren der Kreise

$M_{e,C} = 1$ <=> e liegt auf Kreis C

Sei N die $E(H)$-Kreis-Inzidenzmatrix, d.h.

Zeilen von N <-> Kanten von H

Spalten von N <-> Inzidenzvektoren der Kreise

$N_{e,C} = 1$ <=> e liegt auf Kreis C

Beachte: jede Spalte von N enthält genau eine 1

=> fraktionale Kreis-Packung = $\pi' \in R^k$ mit $\pi' \geq 0$, $M\pi' \leq \mathbf{1}$, $N\pi' = \mathbf{1}$

Führe Schlupfvariablen ein um ein Gleichungssystem zu erhalten und bezeichne den erweiterten Vektor mit π

=> fraktionale Kreis-Packung = $\pi \in R^{k+m}$ ($m = |E(G)|$) mit $\pi \geq 0$, $M\pi = \mathbf{1}$, $N\pi = \mathbf{1}$

$$A\pi = 1,\ \pi \geq 0 \qquad \text{mit } A = \left(\begin{array}{c|c} M & I \\ \hline N & 0 \end{array}\right)$$

d.h. $\mathbf{1}$ in $C(A_1, \ldots, A_{k+m})$ mit A_j = Spalte von A

Anwendung des Lemma von Farkas ergibt Bedingung (3)

Aus dem Farkas Lemma folgt: es gibt ein solches π

<=> für alle $y \in R^{|E(G)|+|E(H)|}$ gilt: $y^T A_j \geq 0$ für alle $j = 1,\ldots,k+m$ => $y^T\mathbf{1} \geq 0$

Teile y auf in $(z,v)^T$, so dass z zu den Zeilen von M (Kanten von G) und v zu den Zeilen von N (Kanten von H) korrespondiert

Für die Spalten A_j zu den Schlupfvariablen folgt:

$y^T A_j \geq 0$ => $z_i \geq 0$

Für die anderen Spalten A_j folgt

$y^T A_j \geq 0$ => $z^T M_j + v^T N_j \geq 0$

Sei C_j der Kreis zu Spalte A_j

=> C_j zerfällt in einen Weg W_j in G und eine Kante f aus H

Dann ist

$z^T M_j$ = Länge $z(W_j)$ des Weges W_j bzgl. Kantenbewertung z(e)

$v^T N_j$ = Kantenwert v(f), wobei f die Kante aus H ist, die auf dem Kreis C_j liegt

Also gilt

$y^T A_j \geq 0$ => $z(W_j) + v(f) \geq 0$

Da dies für alle Kreise C_j gilt, die die Kante f enthalten, ist $z(W_j) + v(f) \geq 0$ äquivalent zu

$dist_{G,z}(s,t) + v(f) \geq 0$ mit $f = \{ s, t \}$ (1)

Die Bedingung $y^T\mathbf{1} \geq 0$ wird zu

$\Sigma_{e \in E(G)} z(e) + \Sigma_{e \in E(H)} v(e) \geq 0$ (2)

Also muss nach dem Farkas Lemma für beliebige (z,v) gelten (3)

$z(e) \geq 0$, $dist_{G,z}(s,t) + v(f) \geq 0$ für alle Kanten $f = \{ s, t \}$ in H

=> $\Sigma_{e \in E(G)} z(e) + \Sigma_{f \in E(H)} v(f) \geq 0$

Bedingung (3) ist äquivalent zum Distanzkriterium (Beweis durch Äquivalenz der Negationen)

(3) verletzt => Distanzkriterium verletzt

(3) verletzt => es gibt z, v mit

$z(e) \geq 0$,

$dist_{G,z}(s,t) + v(f) \geq 0$ für alle Kanten $f = \{ s, t \}$ in H und

$\Sigma_{e \in E(G)} z(e) + \Sigma_{f \in E(H)} v(f) < 0$

$\Rightarrow 0 < \Sigma_{f \in E(H)} dist_{G,z}(s,t) + \Sigma_{f \in E(H)} v(f) < \Sigma_{f \in E(H)} dist_{G,z}(s,t) - \Sigma_{e \in E(G)} z(e)$

$\Rightarrow \Sigma_{e \in E(G)} z(e) < \Sigma_{f \in E(H)} dist_{G,z}(s,t)$

$\Rightarrow$ Distanzkriterium verletzt

Distanzkriterium verletzt $\Rightarrow$ (3) verletzt

Distanzkriterium verletzt

$\Rightarrow$ es gibt $z \geq 0$ mit $\Sigma_{e \in E(G)} z(e) < \Sigma_{f \in E(H)} dist_{G,z}(s,t)$

wähle $v(f) := - dist_{G,z}(s,t)$ für Kante $f = \{s, t\}$ in H

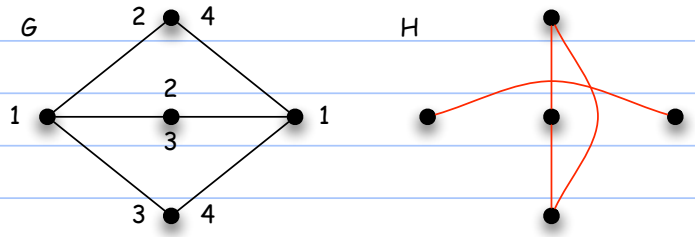
$\Rightarrow dist_{G,z}(s,t) + v(f) \geq 0$ für alle Kanten $f = \{ s, t \}$ in H und

$\Sigma_{e \in E(G)} z(e) + \Sigma_{f \in E(H)} v(f) = \Sigma_{e \in E(G)} z(e) - \Sigma_{f \in E(H)} dist_{G,z}(s,t) < 0$

$\Rightarrow$ (3) ist verletzt ❑

Das Distanzkriterium ist schärfer als das Schnittkriterium

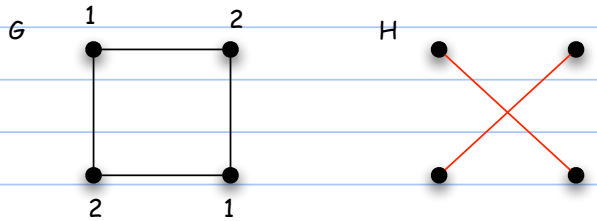
Beispiel 4.6 erfüllt nicht das Distanzkriterium



Setze $z(e) = 1$ für alle e in G $\Rightarrow \Sigma_{f = \{s,t\} \in E(H)} dist_{G,z}(s,t) = 8$, $\Sigma_{e \in E(G)} z(e) = 6$

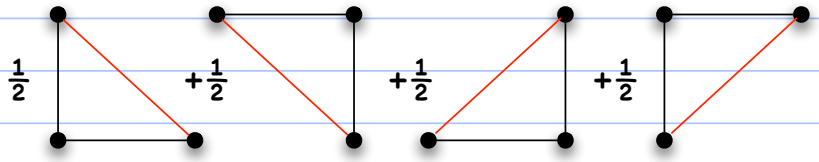
Das Distanzkriterium ist nicht hinreichend für die Existenz disjunkter Wege

Eine Instanz des Disjunkte-Wege-Problems



Eine fraktionale Kreis-Packung

4.4 Das Farkas Lemma

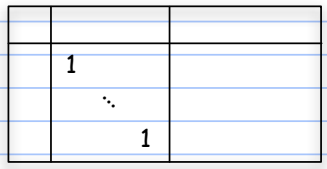


- Wegen Satz 4.7 ist das Distanzkriterium erfüllt.
- Das Disjunkte-Wege-Problem hat keine Lösung

4.5 Duale Information im Tableau

- Wie kann man aus dem optimalen primalen Tableau duale Information bekommen?
  - Seien o.B.d.A. im Ausgangstableau (ggf. mit künstlichen Variablen aus Phase I) die ersten m Spalten die Basisspalten

    => diese sind auf Einheitsmatrix transformiert



- Im optimalen Tableau zur Basis B gilt
  - die Zeilen 1,...,m entsprechen dem Ausgangstableau multipliziert von links mit B^{-1}
  - die reduzierten Kosten ergeben sich gemäß

    $\bar{c}_j = c_j - \pi^T A_j \geq 0$ (4.10)

    wobei π eine Optimallösung des dualen Problems ist (Beweis starker Dualitätssatz)
  - Für die Spalten 1,...,m (Einheitsvektoren im Ausgangstableau) ergibt sich daher

    $\bar{c}_j = c_j - \pi^T A_j = c_j - \pi_j$ (4.11)

Also erhält man eine optimale duale Lösung aus dem optimalen Tableau gemäß

$\pi_j = c_j - \bar{c}_j \; (j = 1, \ldots, m)$ (4.12)

Beachte: dies bezieht sich nur auf das duale Problem zum Ausgangstableau (und nicht auf duale Versionen zu anderen, äquivalenten primalen Formulierungen).

Ferner steht in den ersten m Spalten gerade $B^{-1} = B^{-1} I$ (4.13)

	$c_j - \pi_j$	
	B^{-1}	

4.8 Beispiel (Fortsetzung vom Beispiel zur Zwei-Phasen-Methode)

Ausgangstableau

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z$	0	0	0	0	1	1	1	1	1
$-\xi$	0	1	1	1	0	0	0	0	0
x_1^a	1	1	0	0	3	2	1	0	0
x_2^a	3	0	1	0	5	1	1	1	0
x_3^a	4	0	0	1	2	2	1	0	1

Optimales Tableau

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z$	-9/2	5/2	-1	-1	3/2	0	3/2	0	0
$-\xi$	0	1	1	1	0	0	0	0	0
x_2	1/2	1/2	0	0	3/2	1	1/2	0	0
x_4	5/2	-1/2	1	0	7/2	0	1/2	1	0
x_5	3/2	-5/2	0	1	-11/2	0	-1/2	0	1

(4.12) ergibt $\pi_1 = 0 - 5/2 = -5/2$

$\pi_2 = 0 - (-1) = 1$

$\pi_3 = 0 - (-1) = 1$

für die Werte der Dualvariablen zum dualen Problem bzgl. der primalen Formulierung mit Hilfsvariablen x_i^a

4.9 Beispiel (Fortsetzung des Beispiel zum Kürzeste-Wege-Problem)

Ausgangstableau hat keine Einheitsmatrix, aber 2 Einheitsvektoren)

=> eine Hilfsvariable für Phase I einführen

			x^a	f_1	f_2	f_3	f_4	f_5
	$-\xi$		1	0	0	0	0	0
	$-z$		0	1	2	2	3	1
s	x^a	1	1	1	1	0	0	0
a	f_4	0	0	-1	0	1	1	0
b	f_5	0	0	0	-1	-1	0	1

Kostenkoeffizienten auf reduzierte Form transformieren für ξ und z (0 für Basisvariable)

			x^a	f_1	f_2	f_3	f_4	f_5
	$-\xi$	-1	0	-1	-1	0	0	0
	$-z$	0	0	4	3	0	0	0
s	x^a	1	1	1	1	0	0	0
a	f_4	0	0	-1	0	1	1	0
b	f_5	0	0	0	-1	-1	0	1

Pivotoperation durchführen

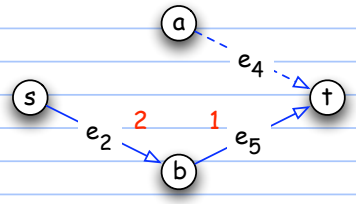
			x^a	f_1	f_2	f_3	f_4	f_5
	$-\xi$	0	1	0	0	0	0	0
	$-z$	-3	-3	1	0	0	0	0
s	f_2	1	1	1	1	0	0	0
a	f_4	0	0	-1	0	1	1	0
b	f_5	1	1	1	0	-1	0	1

=> $\xi = 0$ und x^a aus Basis

=> optimal bzgl. z

Basisspalten im Ausgangstableau

primale Information

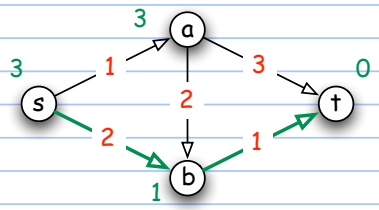


die primale Lösung gibt die Kanten auf dem kürzesten Weg an

duale Information

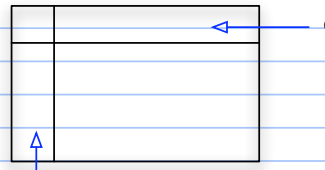
$$\pi_s = c_{x^a} - \bar{c}_{x^a} = 0 - (-3) = 3$$
$$\pi_a = c_4 - \bar{c}_4 = 3 - 0 = 3$$
$$\pi_b = c_5 - \bar{c}_5 = 1 - 0 = 1$$
$$\pi_t = 0$$

$\pi_t = 0$ ergibt sich, da die t-Zeile im primalen LP nicht auftritt



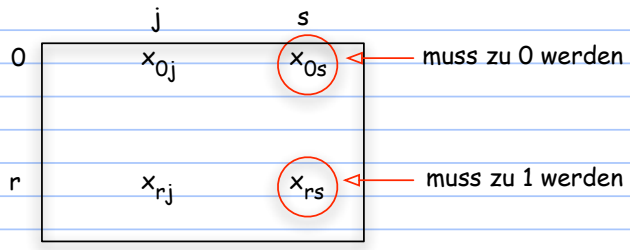
die duale Lösung gibt die kürzeste Entfernung bis zum Knoten t an

- Ziel: das primale Tableau nutzen um das duale LP zu lösen ...
- Kennzeichen des dualen LP
  - bei der Herleitung des dualen LP wurde die primale Optimalitäts-Bedingung $\bar{c} \geq 0$ als duale Restriktion interpretiert
  - => der primale Simplexalgorithmus hat in jeder Pivotoperation eine primal zulässige Lösung
    erst im Optimum ist die duale Restriktion $\bar{c} \geq 0$ erfüllt
- Symmetrische Vorgehensweise für den dualen Simplexalgorithmus
  - erzeuge eine Folge dual zulässiger Lösungen
  - erst im Optimum wird eine primal zulässige Lösung erzeugt
- Herleitung der Operationen im Tableau
  - Tableau X mit Basislösung



- Wähle Pivotzeile r (statt Spalte) mit $x_{r0} < 0$ (d.h. unzulässige Komponente in "primaler" Lösung)
- Zur Wahl der Pivotspalte betrachte in Zeile r die Einträge mit $x_{rj} < 0$ (um nach dem Pivotisieren $x_{r0} \geq 0$ zu erreichen)
  - Pivotisieren mit $x_{rs} < 0$ ergibt in der Kostenzeile

$$x'_{0j} = x_{0j} - \frac{x_{rj}}{x_{rs}} x_{0s} \qquad j = 1, \ldots, n$$



Um dual zulässig zu bleiben, muss $x_{0j}' \geq 0$ bleiben

$$\Rightarrow \frac{x_{0j}}{x_{rj}} \leq \frac{x_{0s}}{x_{rs}} \quad \text{für } x_{rj} < 0$$

=> wähle Spalte s so, dass

$$\frac{x_{0s}}{x_{rj}} = \max \{ \frac{x_{0j}}{x_{rj}} \mid x_{rj} < 0,\ j = 1, \ldots, n \}$$

Beachte die Symmetrie zum primalen Simplexalgorithmus

speziell: alle $x_{rj} > 0$ => duales LP hat unbeschränkte Zielfunktion

4.10 Satz (Interpretation dualer Simplexalgorithmus)

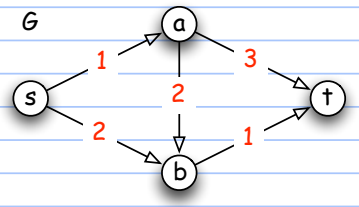
Der duale Simplexalgorithmus entspricht dem primalen Simplexalgorithmus angewendet auf die primale Formulierung des dualen LP

Beweis: Nachrechnen ❑

4.11 Beispiel (Fortsetzung des Beispiel zum Kürzeste-Wege-Problem)

Ausgangstableau, noch nicht auf Basis transformiert und Graph mit Kosten

		f_1	f_2	f_3	f_4	f_5
		1	2	2	3	1
s	1	1	1	0	0	0
a	0	-1	0	1	1	0
b	0	0	-1	-1	0	1

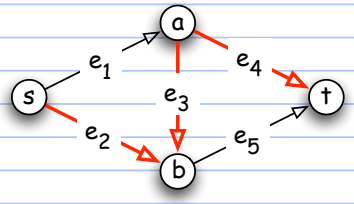


Wähle { 2, 4, 3 } als Basis und transformiere Tableau auf Basisform, stelle Basislösung im Graphen dar

	f_1	f_2	f_3	f_4	f_5
	-3	1	0	0	0	0
f_2	1	1	1	0	0	0
f_4	1	0	0	0	1	1
f_3	-1	-1	0	1	0	-1

=> dual zulässig

nicht primal zulässig



Basislösung entspricht dem s,t-Schnitt $X = \{ s, a \}$

ADM I: s,t-Schnitte sind duale Konfigurationen zu s,t-Flüssen, dies wird über das LP durch die duale Zulässigkeit bestätigt

Ermittlung des Pivotelements

$r = 3$ ist Pivotzeile

Ermittlung der Pivotspalte:

$$\frac{x_{0s}}{x_{rj}} = \max\left\{ \frac{x_{0j}}{x_{rj}} \mid x_{rj} < 0,\ j = 1,\dots,n \right\} = \max\left\{ \frac{1}{-1}, \frac{0}{-1} \right\} = 0$$

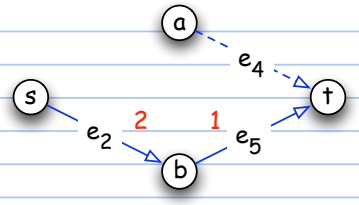
$j = 1$ $\qquad$ $j = 5$

=> $s = 5$ ist Pivotspalte

	f_1	f_2	f_3	f_4	f_5
	-3	1	0	0	0	0
f_2	1	1	1	0	0	0
f_4	1	0	0	0	1	1
f_3	-1	-1	0	1	0	-1

Pivotoperation

	f_1	f_2	f_3	f_4	f_5
	-3	1	0	0	0	0
f_2	1	1	1	0	0	0
f_4	0	-1	0	1	1	0
f_5	1	1	0	-1	0	1



=> primal und dual zulässig => optimal

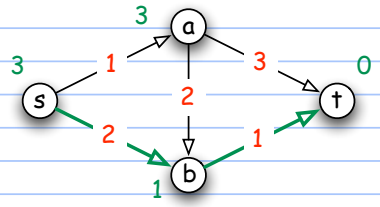
Die duale Optimallösung lässt sich aus der Basisinversen gemäß $\pi^T = c_B^T B^{-1}$ (Dualitätssatz) folgendermaßen berechnen

Optimale Basis ist $B = \{ 2, 4, 5 \}$ mit

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

Also ist

$$\pi^T = c_B^T B^{-1} = (2,3,1) \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} = (3,3,1)$$



Beachte: hier können π bzw. B^{-1} nicht direkt aus dem Tableau abgelesen werden, da das duale LP nicht aus dem Ausgangstableau zur Basis { 2, 4, 3 } gewonnen wurde, sondern das duale LP zu Beispiel 4.9 ist.

5. Berechnungsaspekte des Simplexalgorithmus 28

- Das bisher benutzte Tableau ist redundant (Einheitsmatrix!) und kann u.U. sehr viele Nichtbasis-Spalten haben ...
  - Kern der verwalteten Information ist die Basisinverse B^{-1} zur momentanen Basis B, aus der sich alles berechnen lässt
  - Das revidierte Simplexverfahren ist Grundlage aller kommerziellen LP-Codes

- Grundidee des revidierten Simplexverfahrens: die CARRY Matrix
  - betrachte Starttableau mit Einheitsmatrix = Startbasis links im Tableau

-z	0 0	$\bar{c}_j$
b	I	

CARRY$^{(0)}$

Für jede spätere Basis B steht dann wegen (4.13) an der Stelle von I die Basisinverse B^{-1}

- nach Iteration ℓ stehen im Tableau in den ersten m+1 Spalten folgende Daten

-z´	$-\pi^T$	
b´	B^{-1}	

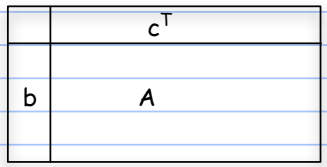
CARRY$^{(\ell)}$

mit

- π^T = duale Lösung wegen (4.12), i.a. unzulässig.

  Die Werte π_i werden auch Simplexmultiplikatoren genannt.
  - b´ = $B^{-1}b$ momentane rechte Seite = aktuelle primale Lösung
  - z´ = $c_B^T B^{-1} b$ momentaner primaler Zielfunktionswert

- Für den Simplexalgorithmus reicht es, folgende Daten zu verwalten:
  1. Ausgangstableau



2. momentane CARRY-Matrix $CARRY^{(\ell)}$

3. momentane Basis durch ihre Spaltenindizes $B(1),\ldots,B(m)$

Hieraus sind alle im Simplexalgorithmus benötigten Daten ableitbar

(1) Pricing Operation (Berechnung der reduzierten Kosten)

berechne

$$\bar{c}_j = c_j - \pi^T A_j$$

nacheinander für Nichtbasis-Variable bis ein reduzierter Kostenkoeffizient $\bar{c}_j < 0$ oder $\bar{c} \geq 0$ (=> Abbruch mit Optimallösung)

(2) Generation of Pivot Column (Erzeugung der Pivotspalte)

berechne

$$X_s = B^{-1} A_s$$

= Spalte s des momentanen Tableaus X

das Pivotelement x_{rs} ergibt sich aus

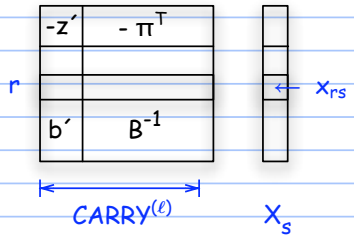
$$\min_{i,\, x_{is} > 0} \frac{b'_i}{x_{is}} \quad \begin{matrix} \leftarrow \text{ in } CARRY^{(\ell)} \\ \leftarrow \text{ in } X_s \end{matrix}$$

oder man stellt fest, dass z unbeschränkt ist (falls alle $x_{is} \leq 0$)

(3) Pivot Operation (Pivotisieren)

berechne $CARRY^{(\ell+1)}$

d.h. transformiere X_s auf den Einheitsvektor (1 beim Pivotelement) und wende die entsprechenden Zeilenoperationen auf $CARRY^{(\ell)}$ an



(4) Basis Update

setze B(r) := s

Bei der Zwei-Phasen-Methode geht man entsprechend vor

man startet mit der künstlichen Kostenfunktion

(1, ..., 1, 0, ..., 0)

künstliche Variable

$x_1^a, \ldots, x_m^a$ bilden Basis

Transformation auf reduzierte Kosten bzgl. dieser Basis ergibt in der Zielfunktionszeile

$$d_j := -\sum_{i=1}^{m} a_{ij}$$

bzgl. der Nichtbasis-Variablen

Am Ende von Phase I erfolgt der Übergang zur ursprünglichen Kostenfunktion

=> $-\pi^T = -c_B^T B^{-1}$ neu berechnen und in $CARRY^{(\ell)}$ übernehmen

$-z = -c_B^T b'$ neu berechnen und in $CARRY^{(\ell)}$ übernehmen

möglich, da c_B in den Ausgangsdaten vorhanden ist, B gemerkt wird und B^{-1} und b' in $CARRY^{(\ell)}$ enthalten sind

(1) Nicht alle Nichtbasis-Spalten pro Iteration behandeln

in der Regel muss man nicht alle reduzierten Kosten

$$\bar{c}_j = c_j - \pi^T A_j$$

berechnen (partial pricing, geht aber nur im primalen LP)

(nur im Extremfall nötig, nur dann gleicher Aufwand wie im vollen Tableau)

(2) Man benutzt in jeder Iteration Spalten A_j des Ausgangstableaus

dieses ist oft dünn besetzt (besonders bei kombinatorischen Problemen, z.B. nur 2 Einträge ≠ 0 bei Kürzeste-Wege-Problem)

=> effiziente Speicher- und Berechnungstechniken verwendbar

(3) $CARRY^{(\ell)}$ nicht explizit berechnen

man braucht nicht ganz $CARRY^{(\ell)}$ zu speichern, da $CARRY^{(\ell)}$ auf einfache Weise aus $CARRY^{(\ell-1)}$ entsteht

$$CARRY^{(\ell)} = P_\ell \cdot CARRY^{(\ell-1)}$$

mit

$P_\ell = (e_1, \ldots, e_{r-1}, \eta, e_{r+1}, \ldots, e_m)$ bildet elementare Zeilenoperation nach

e_i = i-ter Einheitsvektor

$$\eta = \begin{pmatrix} -\frac{x_{1s}}{x_{rs}} \\ \vdots \\ \frac{1}{x_{rs}} \\ \vdots \\ -\frac{x_{ms}}{x_{rs}} \end{pmatrix} \leftarrow \text{Pivotzeile}$$

$\Rightarrow$ P_ℓ speichern durch η-Vektor und Position r

induktiv

$$CARRY^{(\ell)} = P_{\ell-1} \cdot P_{\ell-2} \cdot \ldots \cdot P_1 \cdot CARRY^{(0)}$$

ferner: falls ℓ groß wird, so kann eine Reinvertierung erfolgen, d.h. die Suche nach einer äquivalenten kürzeren Folge von η-Vektoren ($\ell \leq m$) reichen

(4) Diese Techniken verbinden mit numerischen Verfahren für numerische Stabilität

LU-Zerlegung, Cholesky-Faktorisierung

(VL Numerische Mathematik)

Jede reguläre Matrix B hat eine Darstellung als $B = P \cdot L \cdot U$ mit

P = Permutationsmatrix

L = untere Dreiecksmatrix

U = obere Dreiecksmatrix

Dann lassen sich Gleichungssysteme $Bx = b$ wegen $LUx = P^T b$ einfach lösen durch

Lösen von $Ly = P^T b$

Lösen von $Ux = y$

d.h. durch 2 Gleichungssysteme in Dreiecksform

Einige Zitate von Bob Bixby, dem "Vater" von Cplex und Gurobi

Bixby
ISMP 2003

aus Solving Real-World Linear Programs: A Decade and More of Progress, Operations Research (50) 2002, 3-15

It was thus around 1987 that I became seriously involved in the computational aspects of linear programming.

The first version of CPLEX, CPLEX 1.0, was released in 1988.

Advances in computing machinery

Table 1: **Machine improvements–Simplex algorithms**

Old machine/processor	New machine/processor	Estimated speedup
Sun 3/50	Compaq Server ES40, 667 MHz	900
Sun 3/50	Pentium 4, 1.7 GHz	800
25 MHz Intel 386	Compaq Server ES40, 667 MHz	400
IBM 3090/108S	Compaq Server ES40, 667 MHz	45
Cray X-MP/416	Compaq Server ES40, 667 MHz	10

Table 2: **Machine improvements–Barrier algorithms**

Old machine/processor	New machine/processor	Estimated speedup
Sun 3/50	Pentium 4, 1.7 GHz	13000
Sun 3/50	Compaq Server ES40, 667 MHz	12000
33 MHz Intel 386	Compaq Server ES40, 667 MHz	4000
IBM 3090/108S	Compaq Server ES40, 667 MHz	10
Cray X-MP/416	Compaq Server ES40, 667 MHz	5

Algorithmic improvements

The dual simplex algorithm with steepest edge.

The dual simplex algorithm was introduced by Lemke [1954]. It is not a new algorithm. However, to my knowledge, commercial implementations of this algorithm were not available in 1987 as full-fledged alternatives to the primal simplex algorithm. [...]

All that has changed. The dual simplex algorithm is now a standard alternative in modern codes. Indeed, computational tests, some of which will be presented later in this paper, indicate that the overall performance of the dual algorithm may be superior to that of the primal algorithm.

There are a number of reasons why implementations of the dual simplex algorithm have become so powerful. The most important is an idea introduced by Goldfarb and Forrest [1992], a so-called "steepest-edge" rule for selecting the "leaving variable" at each dual simplex iteration. This method requires relatively little additional computational effort per iteration and is far superior to "standard" dual methods, in which the selection of the leaving variable is based only upon selecting a basic variable with large primal infeasibility.

Linear algebra

Linear algebra improvements touch all the parts of simplex algorithms and are also crucial to good implementations of barrier algorithms. Enumerating all such improvements is beyond the scope of this paper. I will mention only a few. For simplex algorithms, two improvements stand out among the rest.

The first of these to be introduced was dynamic LU-factorization using Markowitz threshold pivoting. This approach was perfected by Suhl and Suhl [1990], and has become a standard part of modern codes. In previous-generation codes, "preassigned pivot" sequences were used in the numerical factorization (see

Hellerman and Rarick [1971]). These methods were very effective when no numerical difficulties occurred, but encountered serious difficulties in the alternative case.

The second major linear algebra improvement is that LP codes now take ad- vantage of certain ideas for solving large, sparse linear systems, ideas that have been known in the linear-algebra community for several years (see Gilbert and Peierls [1988]). At each major iteration of a simplex algorithm, several size- able linear systems must be solved. The order of these systems is equal to the number of constraints in the given LP. Typically these systems take as input a vector with a very small number of nonzero entries, say between one and ten – independent of overall model size – and output a vector with only a few additional nonzeros. Since it is unlikely that the sparsity of the output is due to cancellation during the solve, it follows that only a small number of nonzeros in the LU-factorization (and update) of the basis could have been touched during the solve. The trick then is to carry out the solve so that the work is linear in this number of entries, and hence, in total, essentially a constant time operation, even as problem size grows. The effect on large linear programs can be enormous.

Presolve

This idea is made up of a set of problem reductions: Removal of redundant constraints, fixed variables, and other extraneous model elements. The seminal reference on this subject is Brearley et al [1975]. Presolve

was available in MPS III, but modern implementations include a much more extensive set of reductions, including so-called aggregation (substituting out variables, such as free variables, the satisfaction of the bounds of which are guaranteed by the satisfaction of the bounds on the variables that remain in the model). The effects on problem size can be very significant, in some cases yielding reductions by factors exceeding an order of magnitude. Modern presolve implementations are seamless in the sense that problem input and solution output occur in terms of the original model.

Examples of performance improvements

Table 10: **Solution times–Best simplex**

Model	CPLEX 1.0	CPLEX 2.2	CPLEX 5.0	CPLEX 7.1	Algorithm
car	1555.0	701.1	275.8	120.6	primal
continent	364.7	110.5	104.4	46.7	primal
energy1	1217.4	275.0	260.5	22.6	dual
energy2	10130.1	736.0	664.0	693.9	dual
energy3	21797.1	271.9	229.1	161.7	dual
fuel	5619.5	1123.2	698.6	675.0	primal
initial	3832.2	102.2	51.3	15.5	dual
schedule	152404.0	252.3	220.8	64.6	dual

full paper

SOLVING REAL-WORLD LINEAR PROGRAMS: A DECADE AND MORE OF PROGRESS

ROBERT E. BIXBY

ILOG, Inc. and Rice University, bixby@ilog.com or bixby@rice.edu

This paper is an invited contribution to the 50th anniversary issue of the journal *Operations Research*, published by the Institute of Operations Research and Management Science (INFORMS). It describes one person's perspective on the development of computational tools for linear programming. The paper begins with a short personal history, followed by historical remarks covering the some 40 years of linear-programming developments that predate my own involvement in this subject. It concludes with a more detailed look at the evolution of computational linear programming since 1987.

1. INTRODUCTION

I am a relative newcomer to computation. For the first half of my scientific career, my research focused exclusively on the theoretical aspects of operations research and discrete mathematics. That focus began to change in the early 1980s with the appearance of personal computers.

My first PC was used primarily to implement elementary algorithms used in teaching. At first these algorithms did not include a simplex algorithm; eventually, however, I concluded that it would be useful to incorporate computation in the LP courses that I was teaching. As a result, I started writing my own code, initially a simple tableau code.

At that time, in the early 1980s, I knew nothing about the computational aspects of linear programming (LP). I knew a great deal of theory, but numerical analysis and the computational issues associated with numerical algorithms were not subjects that were part of my graduate education. I had no idea that tableaus were numerically unstable.

Fortunately for me, by the time my interests in computation had started, the Department of Industrial Engineering and Management Sciences at Northwestern University had hired Bob Fourer, one of the creators of the AMPL modeling language. Bob had worked for several years at the National Bureau of Economic Research doing practical linear programming, followed by a graduate career at Stanford. He knew a lot about the computational aspects of mathematical programming, and he passed on a great deal of that knowledge to me in informal conversations.

Linear programming become more central to what I was doing when a friend of mine, Tom Baker, founded Chesapeake Decision Sciences (now a part of Aspen Technologies). Shortly thereafter, Tom asked if I had an LP code that he could use in the LP module of the product he was building. I said yes, converted my code to C (that was one of Tom's conditions), and delivered it to him.

To this day, I'm not quite sure why Tom thought my code would eventually be reasonably good. Initially it certainly was not.

After the code was delivered to Chesapeake, there followed a period of about two years during which I received a steady stream of practical LPs from Chesapeake, LPs on which my code did not do very well. In each case, I poked around in my code and the LP itself to see what ideas I could come up with, never looking in the literature (this wasn't my area of research). Slowly the code got better, until some time around 1986, one of Tom's colleagues informed me that my code had actually gotten good enough that one of their customers was interested in obtaining it separately. I was, to say the least, surprised, and immediately set about doing my first actual comparisons to other LP codes. I chose Roy Marsten's (1981) quite successful and portable (that was key for me) XMP code. I discovered, to my amazement, that for a substantial subset of the *netlib*[1] testset my code was indeed pretty good, running on average two times faster than XMP. In addition, it appeared that my code was significantly more stable than XMP.

This comparison to XMP was an important part of what transformed LP computation into a serious part of my scientific research. Equally important was integer programming.

This was the mid-1980s, and integer-programming computational research was beginning to flower, with important contributions by people such as Martin Grötschel, Ellis Johnson, Manfred Padberg, and Laurence Wolsey. Linear programming was an essential component in that work, but the tools available at that time were proving to be inadequate. The then state-of-the-art codes, such as MPSX/370, simply were not built for this kind of application; in addition, they did not deal well with issues such as degeneracy. The situation at the time is well described by some remarks of Grötschel and Holland (1991), commenting on their use of MPSX/370 in work on the traveling salesman problem: They note that if the LP-package they were using had been

Subject classification: Professional: comments on
Area of review: Anniversary Issue (Special).

0030-364X/02/5001-0003 $05.00
1526-5463 electronic ISSN

Operations Research © 2002 INFORMS
Vol. 50, No. 1, January–February 2002, pp. 3–15

bixby-or2002.pdf

- Ziele dieses Abschnittes
  - Illustration des revidierten Simplexalgorithmus
  - Illustration des Umgangs mit LPs mit (exponenziell) vielen Spalten: Column Generation
  - Als Beispiel dafür dient eine Pfad-basierte Formulierung des Max-Fluss-Problem

- Das Max-Fluss-Problem (vergleiche ADM I)
  - Maximum Flow Problem (MFP)
    - Instanz
      - Netzwerk (G, u, s, t) mit

        G Digraph

        s, t Knoten von G

        $u(e)$ = Kapazität der Kante e
    - Aufgabe
      - Finde s,t-Fluss f mit maximalem Wert $v(f)$

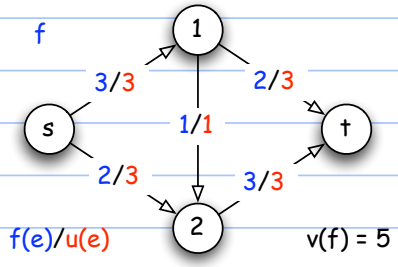
        s,t-Fluss f = Kantenbewertung $f(e)$ mit

        $0 \le f(e) \le u(e)$ für alle Kanten e

Flusserhaltung in allen Knoten $v \neq s, t$

Wert $v(f)$ = Nettoausfluss aus der Quelle s

- Beispiel: ein Fluss f



- Eine Formulierung des Max-Fluss-Problem als LP über Kanten-Variable (Kanten-basierte Formulierung)
  - wie beim Kürzeste-Wege-Problem über Knoten-Kanten-Inzidenzmatrix

max v (maximiere den Flusswert v)

$$Af = b \text{ mit } b = \begin{pmatrix} +v \\ -v \\ 0 \\ \vdots \\ 0 \end{pmatrix} \begin{matrix} \text{Zeile } s \\ \text{Zeile } t \text{ (redundant)} \\ \\ \\ \text{Flusserhaltung} \end{matrix}$$

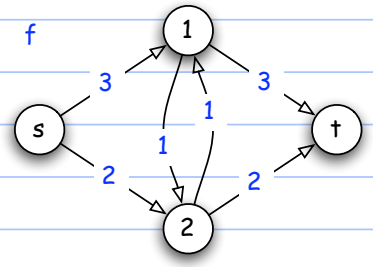
$f \leq u$

$f \geq 0$

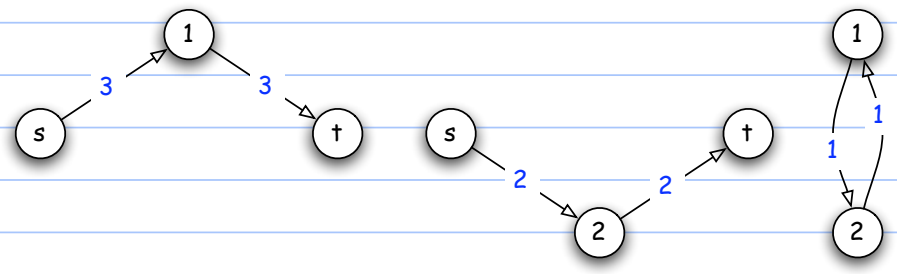
Dabei ist $E(G) = \{ e_1, \ldots, e_m \}$ und $f = (f_1, \ldots, f_m)^T$, d.h., wir haben eine Variable pro Kante, die den Flusswert auf der Kante angibt

- Eine Formulierung des Max-Fluss-Problem als LP über Weg-Variable (Weg-basierte Formulierung)
  - beruht auf dem Fluss-Dekompositionssatz der ADM I
  - ADM I, Satz 5.2 (Dekompositionssatz für s,t-Flüsse, Ford-Fulkerson 1962)
    - Sei $f \neq 0$ ein s,t-Fluss in (G, u, s, t). Dann gilt
      - f ist positive Linearkombination von (Inzidenzvektoren von) gerichteten (elementaren) s,t-Wegen und gerichteten (elementaren) Kreisen
      - die Anzahl der Wege und Kreise kann auf höchstens m beschränkt werden

      - ist f ganzzahlig, so können die Koeffizienten in der Linearkombination als ganzzahlig gewählt werden
  - Beispiel
    - s,t-Fluss



    - Zerlegung in gerichtete Wege und Kreise (nicht eindeutig)



- zugehörige Linearkombination
  - $$\begin{pmatrix} f_{s1} \\ f_{s2} \\ f_{12} \\ f_{1t} \\ f_{21} \\ f_{2t} \end{pmatrix} \quad f = \begin{pmatrix} 3 \\ 2 \\ 1 \\ 3 \\ 1 \\ 2 \end{pmatrix} = 3 \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} + 2 \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} + 1 \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$
- Beachte: Für den Flusswert sind nur die Wege wichtig, Kreise spielen keine Rolle
- Das Weg-basierte LP
  - Seien $W_1, \ldots, W_p$ alle gerichteten elementaren s,t-Wege in G (beachte: p kann exponenziell in n sein).

    Die Kanten-Wege-Inzidenzmatrix $D = (d_{ij})$ ist definiert durch

    $$d_{ij} := \begin{cases} 1 & \text{Kante } e_i \text{ liegt auf Weg } C_j \quad i = 1, \ldots, m \\ 0 & \text{sonst} \qquad\qquad\qquad\qquad\quad j = 1, \ldots, p \end{cases}$$

    $f = (f_1, \ldots, f_p)^T$ ist ein Flussvektor, der pro s,t-Weg W_j eine Komponente f_j enthält. Diese gibt gemäß Dekompositionssatz den Betrag des über den Weg W_j geschickten Flusses an.

- Die Kapazitätsbedingungen lauten

  $$Df \leq u$$

  d.h. pro Zeile e_i gilt

  $$(d_{i1}, \ldots, d_{ip}) \cdot \begin{pmatrix} f_1 \\ \vdots \\ f_p \end{pmatrix} \leq u_i$$

  Σ der Flüsse in allen Wegen, die e_i enthalten
  = Kantenfluss in Kante e_i
- Die Flusserhaltung ist trivialerweise erfüllt und der Flusswert v ergibt sich als

  $$v = \Sigma_j f_j$$
- Das Weg-basierte LP lautet dann

  $$\min c^T f \text{ mit } c_j = -1$$

  $$Df \leq u$$

  $$f \geq 0$$
- In Standardform bringen durch Schlupfvariable s_i

=> $\min\ c'^{T} f' \quad \text{mit } f' = (f \mid s)^T,\ c' = (c \mid 0)^T$

$D' f' = u \quad \text{mit } D' = (D \mid I)$ (5.1)

$f' \geq 0$

Dabei drückt die Schlupfvariable s_i die Restkapazität auf der Kante e_i aus

Lösung der Weg-basierten Formulierung mit dem revidierten Simplexverfahren

Zulässige Basislösung ist gegeben durch $f = 0$ und $s = u$

=> Phase I ist nicht notwendig

die Spalte D_j zum Weg W_j hat (nach Pricing Rule) reduzierte Kosten $\bar{c}_j < 0$

$\Leftrightarrow \quad \bar{c}_j = c_j - \pi^T D_j < 0$

$\Leftrightarrow \quad (-\pi)^T D_j < 1 \quad \text{wegen } c_j = -1$

Interpretation

$-\pi$ = Gewichtung der Kanten e_i mit $-\pi_i$

$(-\pi)^T D_j$ = Länge des Weges W_j bzgl. Gewichtung der Kanten mit $-\pi_i$

Idee

Berechne kürzesten s,t-Weg W bzgl. $-\pi$. Dann gilt (5.2)

W hat Länge ≥ 1 => Optimalitätskriterium ist für alle Spalten erfüllt, die zu Wegen gehören

W hat Länge < 1 => Spalte (Weg) für Basisaustausch gefunden

Man braucht also nicht alle (exponenziell vielen) Spalten zu Wegen explizit zu erzeugen

Im Allgemeinen ist diese Idee

Finde eine Spalte, die das Optimalitätskriterium am meisten verletzt

wieder ein LP, dessen Lösung sehr oft einfacher ist als alle Spalten des revidierten Simplexverfahrens explizit zu erzeugen.

Dies nennt man Column Generation

Für die Spalten zu den Schlupfvariablen s_i gilt

s_i hat negative reduzierte Kosten $\Leftrightarrow -\pi_i < 0$ (5.3)

denn:

reduzierte Kosten zu Spalte s_i ergeben sich als $0 - (\pi^T I)_i = -\pi_i < 0$

5.1 Satz (Lösung des Pfad-basierten Max-Fluss-Problem mit Column Generation)

Eine zulässige Basislösung für (5.1) erfüllt das Optimalitätskriterium

<=> $-\pi \geq 0$ und der kürzeste s,t-Weg bzgl. $-\pi$ hat eine Länge ≥ 1

Die Simplexschritte im revidierten Simplexverfahren entsprechen Kürzeste-Wege-Berechnungen bzgl. $-\pi \geq 0$ bzw. der Aufnahme einer Schlupfvariablen s_i in die Basis falls $-\pi_i < 0$

Beweis

Ist $-\pi_i < 0$ für ein i, so ist s_i wegen (5.3) eine Nichtbasis-Variable mit negativen reduzierten Kosten => Optimailtätskriterium nicht erfüllt

Ist $-\pi \geq 0$, so reduziert sich das Pricing wegen (5.2) und (5.3) auf die Berechnung eines kürzesten s,t-Weges mit nicht-negativer Kantenbewertung $-\pi$. ❑

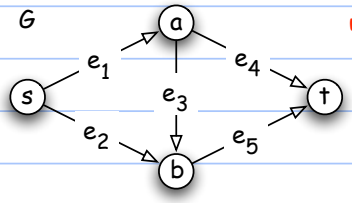
Folgerung

Der revidierte Simplexalgorithmus löst das Max-Fluss-Problem (im Wesentlichen) als eine Folge von Kürzeste-Wege-Problemen

In jeder Iteration muss nur eine (m+1)x(m+1) CARRY Matrix verwaltet werden

5.1 Beispiel

Daten bereitstellen für revidierten Simplexalgorithmus



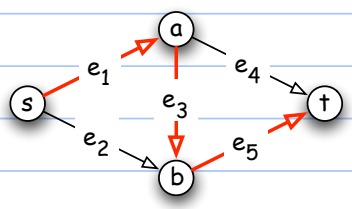
$u_i = 1$

$CARRY^{(0)}$

		0	0	0	0	0	0	← $-\pi^T$
s_1	1	1						
s_2	1		1					
s_3	1			1				
s_4	1				1			
s_5	1					1		

$-\pi^T = 0$ => jeder s,t-Weg ist kürzester Weg mit Kosten 0 < 1

=> wähle o.B.d.A. $W_1 = \{ e_1, e_3, e_5 \}$ als neue Spalte



Die zugehörige Spalte D_1 des Ausgangsproblem ist $(1, 0, 1, 0, 1)^T$ und als transformierte Spalte X_1

erhält man

$$X_1 = B^{-1}D_1 = D_1$$

Der reduzierte Kostenkoeffizienten zur Spalte X_1 ist nach den Vorüberlegungen gerade

$$c_1 - \pi^T D_1 = -1 + \text{Länge des Weges} = -1 + 0 = -1$$

Daten für Pivotoperation:

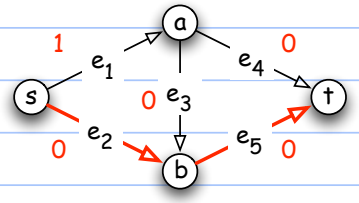
	0	0	0	0	0	0	−1
s_1	1	1					1
s_2	1		1				0
s_3	1			1			1
s_4	1				1		0
s_5	1					1	1

1. Pivotoperation

CARRY(1)

	1	1	0	0	0	0
f_1	1	1				
s_2	1		1			
s_3	0	−1		1		
s_4	1				1	
s_5	0	−1				1

η-Vektor



mit $-\pi$ gewichteter Graph
und kürzester Weg

Die zugehörige Spalte D_2 des Ausgangsproblem ist $(0, 1, 0, 0, 1)^T$ und als transformierte Spalte X_2 erhält man

$$X_2 = B^{-1}D_2 = D_2$$

Als reduzierte Kosten ergeben sich c_2 + Länge kürzester Weg = −1 + 0 = −1

Daten für nächste Pivotoperation:

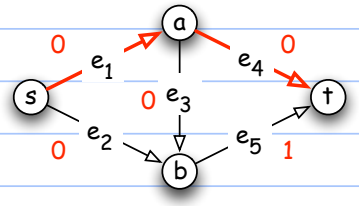
	1	1	0	0	0	0	-1
f_1	1	1					0
s_2	1		1				1
s_3	0	-1		1			0
s_4	1				1		0
s_5	0	-1				1	1

2. Pivotoperation

CARRY$^{(2)}$

	1	0	0	0	0	1
f_1	1	1				
s_2	1	1	1			-1
s_3	0	-1		1		
s_4	1				1	
f_2	0	-1				1

η-Vektor



mit -π gewichteter Graph
und kürzester Weg

Die zugehörige Spalte D_3 des Ausgangsproblem ist $(1, 0, 0, 1, 0)^T$ und als transformierte Spalte X_2 erhält man

$X_3 = B^{-1}D_3 = (1, 1, -1, 1, -1)^T$

Als reduzierte Kosten ergeben sich c_3 + Länge kürzester Weg = -1 + 0 = -1

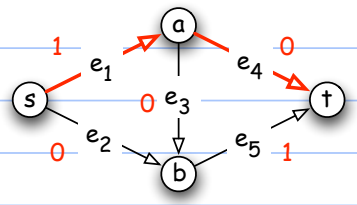
Daten für nächste Pivotoperation:

	1	0	0	0	0	1	-1
f_1	1	1					1
s_2	1	1	1			-1	1
s_3	0	-1		1			-1
s_4	1				1		1
f_2	0	-1				1	-1

3. Pivotoperation

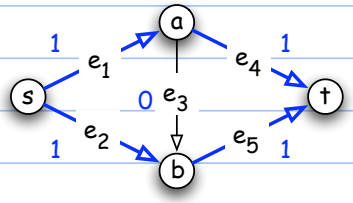
CARRY$^{(3)}$

	2	1	0	0	0	1
f_3	1	1				
s_2	0	0	1			-1
s_3	1	0		1		
s_4	0	-1			1	
f_2	1	0				1



Länge kürzester Weg ≥ 1

=> Optimum erreicht, maximaler Fluss ergibt sich als $f_3 + f_2$



Maximaler Fluss

- Bemerkungen
  - Man verwaltet beim Column Generation also nur ein Teilproblem (genannt Master Problem) mit einigen Spalten und erzeut neue bei Bedarf, indem man das Pricing als gesondertes Optimierungsproblem über alle möglichen Spalten auffasst.
  - Im Beispiel Max-Fluss führt das Pricing auf ein einfaches Kürzeste-Wege-Problem. I.A. ist das Pricing schwieriger und oft sogar NP-schwer (z.B. wenn die Fluss führenden Wege zusätzlichen Bedingungen genügen sollen, etwa in Verkehrsanwendungen: geographisch nicht zu lang, nur Hauptstraßen, ...).
  - Column Generation ist eine der Haupttechniken zur Lösung von komplexen Netzwerkproblemen. Mehr dazu in ADM III

- Ziel dieses Abschnittes
  - den Simplexalgorithmus so abwandeln, dass untere und obere Schranken implizit behandelt werden können
  - dient als Vorbereitung für den Netzwerksimplex (Programmierübung)
  - untere Schranken sind einfach zu behandeln
  - obere Schranken benötigen eine Abwandlung der Definition von zulässiger Basislösung, damit ergibt sich letztlich eine leichte Variation des Simplexalgorithmus

- LP in Standardform mit unteren und oberen Schranken
  - $\min c^T x$ unter $Ax = b$, $\ell \le x \le u$

$$\ell = \begin{pmatrix} \ell_1 \\ \vdots \\ \ell_n \end{pmatrix} \ge 0$$ Vektor von unteren Schranken

$$u = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \ge \ell$$ Vektor von oberen Schranken

Lösungsansätze

(1) Schranken als zusätzliche Restriktionen ansehen und auf Standardform mit Schlupfvariablen bringen

=> Vergrößerung der Matrix A

unerwünscht, da Schranken sehr einfache Restriktionen sind

(2) Schranken implizit berücksichtigen durch leichte Variation des Simplexalgorithmus

-> in diesem Abschnitt

Untere Schranken berücksichtigen durch Transformation der Variablen

Ansatz: $x_j = y_j + \ell_j$

=> $y_j = x_j - \ell_j$

=> Ausgangs-LP schreiben als

$\min c^T y$ unter $Ay = b'$, $0 \le y \le u'$

mit $b' := b - A\ell$ und $u' := u - \ell$

Aus der Lösung y kann dann die Lösung x für das Ausgangs-LP durch

$x = y + \ell$

berechnet werden

Also ab jetzt o.B.d.A. $\ell = 0$ annehmen

d.h. $\min c^T x$

unter $Ax = b$ (5.4)

$0 \le x \le u$

Zulässige Basislösungen für LPs mit oberen Schranken

Erweiterte Partition der Variablen

Statt Partition der Variablen/Spalten von A in B (Basisvariablen) und N (Nichtbasisvariablen) jetzt eine erweiterte Partition in

B Basisvariable

L Nichtbasisvariable mit Wert = untere Schranke = 0

U Nichtbasisvariable mit Wert = obere Schranke = u_j

Für eine solche Partition soll gelten (o.B.d.A. $rang(A) = m$)

$Bx_B + Lx_L + Ux_U = Bx_B + Uu_U = b$ (5.5)

woraus folgt

$x_B = B^{-1}(b - Uu_U) = B^{-1}b - B^{-1}Uu_U$

Eine zulässige Basislösung bzgl. eines LP mit oberen Schranken zur Basis B ist jede Lösung der Form (5.5)

5.2 Satz (Hauptsatz für LPs mit oberen Schranken)

Hat das LP (5.4) eine optimale Lösung, so auch eine optimale Lösung, die zulässige Basislösung der Form (5.5) ist.

Beweis

Modelliere $x \le u$ mit Schlupfvariablen s als Gleichungssystem $x + s = u$

Dann ergibt sich folgendes großes LP in Standardform (LPSU)

$\min\ c^T x$

unter $Ax = b$

$x + s = u$

$x \ge 0,\ s \ge 0$

Da A vollen Zeilenrang hat, hat auch das große LP vollen Zeilenrang.

Sei $(x,s)^T$ eine zulässige Basislösung von (LPSU) zur Basis B'.

Setze

$B := \{ j \in \{1,\dots,n\} \mid x_j \in B' \text{ und } s_j \in B' \}$ und $p := |B|$

$L := \{ j \in \{1,\dots,n\} \mid x_j \notin B' \text{ und } s_j \in B' \}$ und $s := |L|$

$U := \{ j \in \{1,\dots,n\} \mid x_j \in B' \text{ und } s_j \notin B' \}$ und $q := |U|$

Mit der Identifizierung $B = A_B$ und $U = A_U$ lässt sich B' so permutieren, dass man folgende Gestalt erhält:

$$PB'Q^T = \begin{pmatrix} B & B_q & 0 & 0 \\ I_p & 0 & I_p & 0 \\ 0 & I_q & 0 & 0 \\ 0 & 0 & 0 & I_s \end{pmatrix} \begin{matrix} \leftarrow \text{Zeilen zu b} \\ \\ \updownarrow \\ \text{Zeilen zu u} \end{matrix}$$

mit Permutationsmatrizen P und Q

Zählen der Zeilen von B' ergibt $m + n = m + p + q + s$

Zählen der Spalten von B' ergibt $m + n = 2p + q + s$

$\Rightarrow p = m \Rightarrow B$ ist eine mxm-Matrix

Laplace-Entwicklung von $\det(B') = \det(PB'Q)$ nach den letzten n Zeilen ergibt $|\det(B')| = |\det(B)|$

B' ist Basis des großen LP $\Rightarrow \det(B') \neq 0 \Rightarrow \det(B) \neq 0$

$\Rightarrow B$ ist Basis von A

Da $B'(x,s)^T = (b,u)^T$, erhält man aus der permutierten Darstellung von B' die Form (5.6)

$Bx_B + Ux_U = b$

$$x_B + s_B = u_B$$
$$x_U = u_U$$
$$s_L = u_L$$

Also definiert die Indexpartition B, L, U eine Basislösung $Bx_B + Ux_U = b$ der Form (5.5)

=> jeder zulässigen Basislösung vom großen LP entspricht eine zulässige Basislösung von (5.4) in der Form (5.5)

=> Behauptung mit dem Hauptsatz der Linearen Optimierung (Satz 3.12) ❑

Optimalitätskriterium für LPs mit oberen Schranken

5.3 Satz (Optimalitätskriterium für LPs mit oberen Schranken)

Eine zulässige Basislösung der Form (5.5) ist optimal

<=> für die reduzierten Kostenkoeffizienten im Tableau gilt

$$\left.\begin{array}{ll} \bar{c}_j = 0 & \text{für } x_j \in B \\ \bar{c}_j \geq 0 & \text{für } x_j \in L \\ \bar{c}_j \leq 0 & \text{für } x_j \in U \end{array}\right\} \quad (5.7)$$

Beweis:

entsprechende Umformung durch Betrachtung des großen LP ❑

Wahl des Pivotelement für LPs mit oberen Schranken

Spaltenauswahl

Spalte A_j mit $j \in L$ und $\bar{c}_j < 0$ oder Spalte A_j mit $j \in U$ und $\bar{c}_j > 0$ (5.8)

Zeilenauswahl

$x_s \in L$ => vergrößere x_s soviel wie möglich

$x_s \in U$ => verkleinere x_s soviel wie möglich

Nutze dazu die Darstellung von $x(\theta)$ gemäß (3.16)

$$x_\ell(\theta) = \begin{cases} y_{B(i)} - \theta x_{is} & \ell = B(i),\ i = 1, \ldots, m \\ \theta & \ell = s \\ 0 & \text{sonst} \end{cases}$$

und berücksichtige, dass $x_s = u_s$ sein kann

$x_{is} < 0$ => $x_{B(i)}$ wird vergrößert

$\Rightarrow$ Schranke $\theta_i = \dfrac{u_i - x_{i0}}{-x_{is}}$ für θ

$x_{is} > 0$ => $x_{B(i)}$ wird verkleinert

$\Rightarrow$ Schranke $\theta_i = \frac{x_{i0}}{x_{is}}$ für θ

(wie beim gewöhnlichem Simplexverfahren)

- Wähle θ als Minimum der θ_i und u_s. Dann gibt es 2 Fälle
  - Das Minimum wird bei u_s angenommen
    - Lasse die Basis unverändert, die Variable von x_s geht von U nach L oder umgekehrt
  - Das Minimum wird bei $\theta = \theta_r$ angenommen
    - Mache eine Pivotoperation mit x_{rs}

      Der neue Wert von x_s ergibt sich als θ (falls $x_s = 0$) bzw. als $u_s - \theta$ (falls $x_s = u_s$)
- Terminierung
  - erfordert Zusatzüberlegungen

- Eine detaillierte Behandlung des Netzwerksimplexalgorithmus erfolgt in der Übung ...
- Netzwerkproblem (Input für den Netzwerksimplex) ...
- Basislösungen des Netzwerkproblems
  - Im Weiteren sei $V = \{ 1,...,n \}$ und G zusammenhängend.

    Zeilen von A addieren sich zu 0

    => o.B.d.A. Zeile zum Knoten 1 streichen, resultierende Matrix wieder mit A bezeichnen

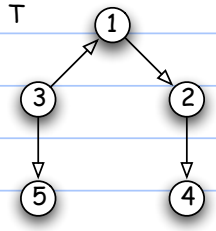
    => A hat $n-1$ Zeilen

  - Betrachte einen spannenden Baum T von G

    => T hat $n-1$ Kanten

    Sei B die Menge der zugehörigen Spalten von A

    Beispiel:



B

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

Fasse T als ungerichteten Baum mit Wurzel 1 auf

Ordne die Knoten von T gemäß Preorder Reihenfolge (d.h. Wurzel-Links-Rechts Reihenfolge)

im Beispiel: 1, 3, 5, 2, 4

Ordne die Kanten gemäß dieser Knotenordnung,

d.h. für jeden Knoten $j \neq 1$ nehme die (eindeutige) letzte Kante auf dem Weg von 1 zu j.

im Beispiel: (3,1) (3,5) (1,2) (2,4)

Betrachte die permutierte Matrix B´ gemäß dieser Zeilen- und Spaltenordnung

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

5.4 Lemma

Die permutierte Matrix B´ gemäß der Preorder Reihenfolge ist (nach Streichung von Zeile 1) eine obere Dreiecksmatrix mit Einträgen ≠ 0 auf der Diagonalen.

Beweis

Die Preorder Reihenfolge besuche gerade den Knoten i.

Sei j der Vater von i in T.

=> (i,j) oder (j,i) ist Baumkante, o.B.d.A. (i,j)

Permutation von Zeilen und Spalten => (i, (i,j)) ist eine Position auf der Diagonalen von B´

Preorder Reihenfolge => j wurde von i besucht

=> Spalte zu (i,j) enthält

-1 in der zu j gehörigen Zeile

+1 in der zu i gehörigen Zeile

0 in allen späteren Zeilen ❑

5.5 Folgerung

Die Zeilen und Spalten der Knoten-Kanten-Inzidenzmatrix eines spannenden Baumes von G können durch die Preorder Reihenfolge so permutiert werden. dass eine nicht-singuläre obere Dreiecksmatrix entsteht.

Die Gleichungssystem $Bx = b$ und $\pi^T B = c_B^T$, die im revidierten Simplexalgorithmus gelöst werden müssen, lassen sich durch die Dreicksform und die Tatsache, dass jede Spalte höchstens 2 Einträge ≠ 0 enthält, sehr einfach durch "iteratives Einsetzen" lösen.

5.6 Satz (Korrespondenz Basis <-> spannender Baum)

Jeder spannende Baum von G definiert eine Basis des Netzwerkproblems (i.A. nicht zulässig bzgl. $0 \le x \le u$).

Jede Basis des Netzwerkproblems definiert einen spannenden Baum von G.

Beweis

=>:

Lemma 5.4.

Insbesondere hat jede Basis n-1 Spalten.

<=:

Sei B eine Basis des Netzwerkproblems

=> die zugehörigen Spalten entsprechen einem Teilgraph G' mit n-1 Kanten

Claim: G' ist kreisfrei

Annahme: G' enthält einen (ungerichteten) Kreis K.

Wähle eine Orientierung von K und sei K^+ die Menge der Vorwärtskanten und K^- die Menge der Rückwärtskanten von K bzgl. der Orientierung.

Jeder Knoten i in K ist genau mit 2 Kanten aus K inzident

$$\Rightarrow \sum_{e \in K^+} A_e - \sum_{e \in K^-} A_e = 0$$

Widerspruch zur Tatsache, dass B eine Basis ist.

ADM I, Satz 2.3 => jeder kreisfreie Graph mit n Knoten und n-1 Kanten ist ein Baum ❑

Schritte im revidierten Simplexalgorithmus

Basis B = Baum gegeben mit Partition B, L, U.

1. Berechnung der rechten Seite

Sie ergibt sich aus

$$Bx_B = b - Uu_U =: b'$$

=> Löse Gleichungssystem $Bx_B = b'$

Durch Rückwärtseinsetzen aus Dreiecksform gemäß Folgerung 5.4

2. Berechnung der Simplexmultiplikatoren π_i

Sie ergeben sich aus

$$\pi^T B = c_B^T$$

=> $\pi_i - \pi_j = c_{ij}$ für die Spalte/Kante $(i,j) \in B$

=> direkt gemäß Dreiecksform von B rückwärts ausrechnen, π_i der letzten Zeile direkt angebbar, dann rückwärts weiter

3. Optimalitätskriterium

reduzierte Kosten der Spalte/Kante (i,j) ergeben sich als

$$\bar{c}_{ij} = c_{ij} - \pi^T A_{ij} = c_{ij} - \pi_i + \pi_j$$

Abschnitt 5.4 =>

$\bar{c}_{ij} \geq 0$ für $(i,j) \in L$

$\bar{c}_{ij} \leq 0$ für $(i,j) \in U$

4. Berechnung der transformierten Spalte X_{rs}

$-X_{rs}$ entspricht der Änderung der Basisvariablen bei Erhöhung des Wertes der Nichtbasisvariablen x_{rs} um 1

(r,s) soll in die Basis, B entspricht Baum

=> B + (r,s) enthält eindeutigen Kreis K

K gemäß (r,s) orientieren ergibt Zerlegung von K in die Menge der Vorwärtskanten und die Menge der Rückwärtskanten.

Änderung von x_{rs} um 1 entspricht Änderung auf K, und zwar

+1 auf Vorwärtskanten

-1 auf Rückwärtskanten

=> Pivotoperation entspricht dem Basistausch über Kantenaustausch auf dem Kreis.

Falls die Nichtbasisvariable x_{rs} den "Engpass" darstellt, so erfolgt gemäß Abschnitt 5.4 nur ein Wechsel von

L nach U bzw. umgekehrt.

- 5. Finden einer zulässigen Ausgangslösung
  - = Phase I, siehe Übung

- 6. Vermeiden von Kreiseln
  - durch Beschränkung auf starke Baumlösungen, siehe Übung.

- Literatur zum Netzwerksimplexalgorithmus
  - Kapitel 11 aus

    K. Ahuja, T.L. Magnanti, J.B. Orlin

    Network Flows: Theory, Algorithms, and Applications

    Prentice Hall, 1993

6. Primal-duale Algorithmen 34

- Hintergrund
  - Die Idee basiert auf der Ausnutzung der Bedingungen vom komplementären Schlupf.
  - Primal duale Algorithmen wurden ursprünglich für Netzwerkprobleme entwickelt [Dantzig, Ford, Fulkerson 1956]
  - Sie stellen auf eine allgemeine Methode dar um "spezialisierte" Algorithmen für Kombinatorische Optimierungsprobleme zu entwerfen, sowohl exakte wie approximative.

- Grundidee
  - Starte mit einem LP in Standardform

    (P) min $z = c^T x$

    $Ax = b \geq 0$ (o.B.d.A.)

    $x \geq 0$
  - Das zugehörige duale LP ist

    (D) max $w = \pi^T b$

    $\pi^T A \leq c^T$

    π nicht vorzeichenbeschränkt

- Bedingungen vom komplementären Schlupf ergeben

  $x \in S_P$, $\pi \in S_D$ sind optimal

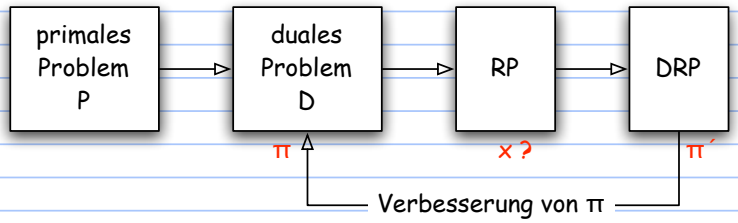
  <=> $\pi_i(a_i^T x - b_i) = 0$ für alle i (erfüllt, da $Ax = b$)

  $(c_j - \pi^T A_j)x_j = 0$ für alle j (6.1)

  Also: (6.1) = einzig verbleibende Bedingungen für Optimalität

- Pimal-dualer Algorithmus

  Zu $\pi \in S_D$ ein $x \in S_P$ suchen, so dass x und π (6.1) erfüllen
  - Die Suche dieses $x \in S_P$ geschieht durch Lösen eines Hilfproblems, des Restricted Primal (RP), das durch die gegebene dual zulässige Lösung $\pi \in S_D$ definiert wird.
  - Falls ein solches x nicht existiert, so nutzt man Informationen über das zu (RP) duale Problem (DRP) um eine "bessere" duale Lösung $\pi \in S_D$ zu konstruieren
  - Dieser Prozess wird dann iteriert bis ein optimales Paar x, π gefunden ist



- Bemerkung: dies ist eigentlich ein dualer Algorithmus, da zu jedem Zeitpunkt eine dual zulässige Lösung π existiert und erst im Optimum eine primal zulässige Lösung x gefunden wird

- Ermittlung einer dual zulässigen Startlösung π
  - alle $c_j \geq 0$
    - => $\pi = 0$ ist dual zulässig, da $\pi^T A \leq c^T$
  - mindestens ein $c_j < 0$
    - Verwende einen Trick:
    - Führe eine weitere primale Variable $x_{n+1} \geq 0$ ein

      Führe eine weitere primale Restriktion

      $$x_1 + x_2 + \ldots + x_{n+1} = b_{m+1}$$

      ein mit $b_{m+1} \geq n \cdot M$ (M aus Lemma 3.4) und $c_{m+1} = 0$

      Lemma 3.4 => diese Restriktion ändert S_P nicht
    - Das duale Problem lautet dann

      $$\max w = \pi^T b + \pi_{m+1} b_{m+1}$$

      $$\pi^T A_j + \pi_{m+1} \leq c_j \quad j = 1,\ldots,n$$

      $$\pi_{m+1} \leq 0$$

      π_i nicht vorzeichenbeschränkt, i = 1,...,m
    - Eine zulässige Lösung dieses dualen LP ist gegeben durch

$\pi_i = 0 \quad i = 1,\dots,m$

$\pi_{m+1} = \min_j c_j < 0$ (da ja mindestens ein $c_j < 0$)

=> eine dual zulässige Lösung ist sehr einfach erzeugbar (wesentlich einfacher als mit Zwei-Phasen-Methode)

Das Restricted Primal (RP)

Nehmen o.B.d.A. an, dass eine dual zulässige Lösung π von (D) gegeben ist

Um (6.1) zu erfüllen, setze

$J := \{ j \mid \pi^T A_j = c_j \}$

Bezeichnung: J = Menge der zulässigen Spalten

(6.1) => $x \in S_P$ ist optimal <=> $x_j = 0$ für alle $j \notin J$

Wir suchen also ein x mit

$\sum_{j \in J} A_j x_j = b$

$x \geq 0, \; x_j = 0$ für alle $j \notin J$

Diese Suche ist ein reines Zulässigkeitsproblem, das mit Phase I des Simplexalgorithmus gelöst werden kann.

Das Phase I Problem wird Restricted Primal (RP) genannt:

$$
\begin{array}{lll}
\min & \xi = \sum_{i=1}^{m} x_i^a & \\
\text{unter} & \sum_{j \in J} a_{ij} x_{ij} + x_i^a = b_i \quad i = 1,\dots,m & \\
& x_j \geq 0 \quad j \in J & \text{(RP)} \\
& x_j = 0 \quad j \notin J & \longleftarrow \text{diese } x_j \text{ kann} \\
& x_i^a \geq 0 \quad i = 1,\dots,m & \text{man weglassen}
\end{array}
$$

(RP) kann mit dem normalen Simplexalgorithmus gelöst werden und entspricht dem Finden einer zulässigen Lösung für (P), wobei die Spalten A_j mit $j \notin J$ weggelassen werden. Eine Anfangsbasis von (RP) ist durch die künstlichen Variablen gegeben.

Falls $\xi_{opt} = 0$, so sind alle künstlichen Variablen 0 und x ist eine Lösung von (RP)

=> x ist optimale Lösung von (P)

Falls $\xi_{opt} > 0$, so existiert keine zulässige Lösung von (RP), in der x (6.1) erfüllt

=> Untersuchung des dualen LP zu (RP)

Das Duale (DRP) des Restricted Primal

(DRP) lautet

$$\left.\begin{array}{lll} \max \quad w = \pi^T b & & (6.2) \\ \text{unter} \quad \pi^T A_j \leq 0 & j \in J & (6.3) \\ \pi_i \leq 1 & i = 1,\ldots,m & (6.4) \\ \pi_i \text{ beliebig} & i = 1,\ldots,m & (6.5) \end{array}\right\} \text{(DRP)}$$

Sei π' eine optimale Lösung von (DRP) (existiert nach Starkem Dualitätssatz)

Idee: kombiniere π' mit der usprünglichen dualen Lösung π zu

$$\pi^* := \pi + \theta\pi' \quad (6.6)$$

wobei θ so gewählt ist, dass π^* zulässig in (D) bleibt und die duale Zielfunktion in (D) echt wächst

Konsequenz für die duale Zielfunktion in (D):

$$(\pi^*)^T b = \pi^T b + \theta(\pi')^T b$$

$\theta(\pi')^T b = \xi_{opt} > 0$ da (RP) und (DRP) ein primal duales Paar bilden

Also muss man $\theta > 0$ wählen damit die duale Zielfunktion echt wächst

Konsequenz für die duale Zulässigkeit in (D)

duale Zulässigkeit bedeutet

$$(\pi^*)^T A_j = \pi^T A_j + \theta(\pi')^T A_j \leq c_j \quad \text{für } j = 1, \ldots, n$$

kein Problem, falls $(\pi')^T A_j \leq 0$ (was für alle $j \in J$ wegen $\pi' \in S_{DRP}$ bereits erfüllt ist)

Es ergeben sich 2 Fälle

$(\pi')^T A_j \leq 0$ für alle $j = 1, \ldots, n$

=> θ kann beliebig groß gewählt werden

=> duale Zielfunktion unbeschränkt

Satz 4.3 => (P) hat keine zulässige Lösung

$(\pi')^T A_j > 0$ für ein $j \notin J$

Dann ergibt sich für θ die Bedingung

$$\pi^T A_j + \theta(\pi')^T A_j \leq c_j$$

$\theta(\pi')^T A_j > 0$

$$\text{also } \theta \leq \frac{c_j - \pi^T A_j}{(\pi')^T A_j}$$

Wir fassen zusammen

6.1 Satz (Unlösbarkeit von (P) im primal-dualen Algorithmus)

Falls $\xi_{opt} > 0$ in (RP) und $(\pi')^T A_j \leq 0$ für alle $j = 1, \ldots, n$ bzgl. der optimalen Lösung π' von (DRP), so hat (P) keine zulässige Lösung.

Beweis: klar nach dem oben Gesagten ❑

6.2 Satz (Verbesserung der dualen Lösung im primal-dualen Algorithmus)

Wenn $\xi_{opt} > 0$ in (RP) und $(\pi')^T A_j > 0$ für ein $j \notin J$, so ist

$$\theta_1 := \min \{ \frac{c_j - \pi^T A_j}{(\pi')^T A_j} \mid j \notin J, (\pi')^T A_j > 0 \} \qquad (6.7)$$

das größte θ, so dass $\pi^* := \pi + \theta\pi'$ dual zulässig bleibt. Dann ist

$$w^* := (\pi^*)^T b = \pi^T b + \theta_1 (\pi')^T b > w$$

Beweis: klar nach dem oben Gesagten ❑

Der primal-duale Algorithmus

Algorithmus (Primal-Dual)

Input

Primales LP (P) in Standardform

Dazu gehöriges duales LP (D) mit zulässiger Lösung π (ggf. über den obigen Trick)

Output

bei Terminierung: Optimale Lösung oder Mitteilung, dass (P) keine zulässige Lösung hat

Terminierung kann durch Auswahlregeln garantiert werden

Methode

repeat

Bestimme (RP) durch Ermittlung von $J := \{ j \mid \pi^T A_j = c_j \}$

call Phase I mit Kostenfunktion $\xi = \sum x_i^a$ für (RP)

if $\xi_{opt} > 0$ then

call dualer Simplex für (DRP) und nehme optimale Lösung π' aus der optimalen Basis

if $(\pi')^T A_j \leq 0$ für alle $j = 1, \ldots, n$

then return "(P) hat keine zulässige Lösung"

else

berechne θ_1 gemäß (6.7)

setze $\pi := \pi + \theta_1 \pi'$

until $\xi_{opt} = 0$

return Lösung x von (RP)

(1) Restart des neuen (RP) mit optimaler Basislösung des vorigen (RP) möglich

6.3 Satz (Erhalt zulässiger Spalten)

Jede zulässige Spalte der optimalen Basis von (RP) bleibt zulässig beim Start der nächsten Iteration des primal-dualen Algorithmus

Beweis

Sei A_j eine zulässige Spalte der optimalen Basis von (RP)

Definition zulässige Spalte => A_j ist Spalte von A, d.h. gehört nicht zu einer Hilfsvariablen

reduzierte Kosten von Basisspalten sind 0, π' duale Optimallösung von (RP)

$\Rightarrow 0 = \bar{c}_j = c_j - (\pi')^T A_j = 0 - (\pi')^T A_j$

$\Rightarrow (\pi')^T A_j = 0$

Dann folgt

$$(\pi^*)^T A_j = \pi^T A_j + \theta_1 (\pi')^T A_j = \pi^T A_j + 0 = \pi^T A_j = c_j$$

da A_j zulässige Spalte bzgl. π

=> A_j bleibt zulässig bzg. π^* ❑

Eine optimale Basis von (RP) setzt sich zusammen aus

zulässigen Spalten => bleiben zulässig wegen Satz 6.3

Spalten zu künstlichen Variablen => bleiben erhalten im neuen (RP)

=> (1)

(2) (RP) kann mit dem revidierten Simplexverfahren gelöst werden

folgt aus Satz 6.3. Man muss nur die Menge J bei den Nichtbasisspalten aktualisieren

(3) Terminierung kann durch Pivotregeln sichergestellt werden

6.4 Satz (Terminierung des primal-dualen Algorithmus)

Der primal-duale ALgorithmus löst (P) in endlich vielen Schritten

Beweis

Fasse (RP) auf als Folge von Pivotisierungen über allen Variablen $x_1^a, \ldots, x_m^a, x_1, \ldots, x_n$

(möglich, da $x_j = 0$ für $r \notin J$ und somit als Nichtbasis-Variable aufgefasst werden kann)

=> (RP) durchläuft eine Folge von zulässigen Basislösungen von $(I \mid A)$

Claim: die Zielfunktion ist bzgl. der Folge schwach monoton fallend

klar innerhalb eines Durchlaufs der repeat-Schleife, da der Algorithmus dann dem normalen (revidierten) Simplexverfahren entspricht.

Betrachte nun den erneuten Eintritt in die repeat-Schleife

=> θ_1 wird ermittelt

Sei r der Index, für den das Minimum bei der Ermittlung von θ_1 angenommen wird

Unter-Claim: $r \in J$ bzgl. des neuen (RP) und Spalte r hat im neuen (RP) negative reduzierte Kosten

$$(\pi^*)^T A_r = \pi^T A_r + \theta_1 (\pi')^T A_r = \pi^T A_r + \frac{c_r - \pi^T A_r}{(\pi')^T A_r} \cdot (\pi')^T A_r = c_r$$

=> A ist zulässige Spalte bzgl. π^* => $r \in J$ bzgl. des neuen (RP)

Spalte r hat im neuen (RP) die reduzierten Kosten (siehe Beweis Satz 6.3)

$0 - (\pi')^T A_r < 0$

da $(\pi')^T A_r > 0$ nach Definition von θ_1

Unter-Claim => beim Wiedereintritt in die repeat-Schleife kann die Spalte r als Pivotspalte gewählt werden im Sinne des normalen Simplexalgorithmus mit schwach monoton fallenden Kosten

Claim => die Anpassung der lexikographische Regel an die Folge von Basislösungen von $(I \mid A)$ stellt die Endlichkeit sicher ❑

Herleitung der verschiedenen LPs (P), (D), (RP), (DRP)

Wir betrachten die Formulierung von (SP) aus Abschnitt 4.3

(P) min $c^T f$

$Af = b$ (A = Knoten-Kanten-Inzidenzmatrix)

$f \geq 0$

mit bereits gestrichener Zeile zum Knoten t

Das duale LP lautet

(D) max $\pi_s - \pi_t$

$\pi_i - \pi_j \leq c_{ij}$ für alle Kanten $(i, j) \in E(G)$

π_i nicht vorzeichenbeschränkt

$\pi_t = 0$ (entspricht der gestrichenen Zeile zu t)

Die Menge der zulässigen Spalten ist

$IJ = \{ (i,j) \in E \mid \pi_i - \pi_j = c_{ij} \}$

Als (RP) ergibt sich

$$\min \ \xi = \sum_{i=1,\dots,n-1} x_i^a$$

$$x^a + Af = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \leftarrow \text{Zeile } s$$

$f_{ij} \geq 0$ für alle Kanten $(i,j) \in E(G)$

$f_{ij} = 0$ für alle Kanten $(i,j) \notin IJ$

$x_i^a \geq 0$ für $i = 1, \dots, n-1$

Das zugehörige duale (DRP) ist

$\max \ w = \pi_s$

$\pi_i - \pi_j \leq 0$ für alle Kanten $(i, j) \in IJ$

$\pi_i \leq 1$ für $i = 1, \dots, n-1$ (ergibt sich aus den Spalten zu den x_i^a)

$\pi_t = 0$

Interpretation des primal-dualen Algorithmus

(1) $\xi_{opt} = 0$ in (RP) <=> es gibt einen Weg von s nach t nur über Kanten aus IJ.

Jeder solche Weg ist Optimallösung für (P), d.h. kürzester s,t-Weg

Beweis

"=>"

Sei $\xi_{opt} = 0$

=> optimale Basislösung von (RP) ist ein s,t-Weg mit $f_{ij} = 0$ für alle Kanten $(i,j) \notin IJ$

=> der Weg benutzt nur Kanten aus IJ.

"<="

Jeder s,t-Weg über Kanten aus IJ ist zulässig in (RP) und hat $\xi = 0$

=> optimal für (P) nach primal dualem Verfahren ❑

(2) Falls kein Weg von s nach t über Kanten aus IJ existiert, so ist π' mit

$$\pi_i' := \begin{cases} 0 & \text{t von i aus über IJ-Kanten erreichbar oder } i = t \\ 1 & \text{sonst} \end{cases}$$

optimal für (DRP)

Beweis

- π' ist zulässig für (DRP)
  - $\pi_i \leq 1$ und $\pi_t = 0$ sind erfüllt.
  - Annahme: $\pi_i' - \pi_j' \leq 0$ ist verletzt für die Kante $(a,b) \in IJ$

    π' hat nur Werte 0 und 1 $\Rightarrow$ $\pi_a' = 1$ und $\pi_b' = 0$

    Definition von π' $\Rightarrow$ t ist von b aus über Kanten aus IJ erreichbar

    $(a,b) \in IJ$ $\Rightarrow$ t ist von a aus über Kanten aus IJ erreichbar

    $\Rightarrow \pi_a' = 0$, Widerspruch
- π' ist optimal für (DRP)
  - Die Zielfunktion ist $\max w = \pi_s$

    Restriktion $\pi_s \leq 1$ $\Rightarrow$ jedes π mit $\pi_s = 1$ ist optimal

    $\Rightarrow$ π' ist optimal ❑

(3) Für $\xi_{opt} > 0$ und π' gemäß (2) definiert ergibt sich

$$\theta_1 = \min \{ c_{ij} - (\pi_i - \pi_j) \mid (i,j) \notin IJ,\ \pi_i' - \pi_j' = 1 \}$$

Beweis

- Sei $\xi_{opt} > 0$ und π' gemäß (2) definiert, also optimal für (DRP)

  Nach (6.7) ist

$$\theta_1 = \min \{ \frac{c_{ij} - \pi^T A_{ij}}{(\pi')^T A_{ij}} \mid (i,j) \notin IJ,\ (\pi')^T A_{ij} > 0 \}$$

$(\pi')^T A_{ij} = \pi_i' - \pi_j' > 0 \Leftrightarrow \pi_i' = 1$ und $\pi_j' = 0$ $\Rightarrow$ $(\pi')^T A_{ij} = \pi_i' - \pi_j' = 1$ ❑

(4) Der primal-duale Algorithmus reduziert (SP) auf eine Sequenz von Erreichbarkeitsproblemen

Ist t von i aus über Kanten aus IJ erreichbar ?

bzw., nach Umdrehen der Kanten,

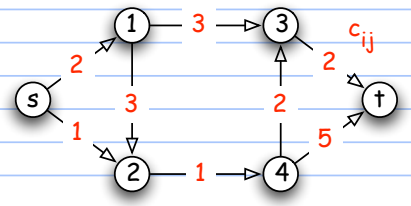
Welche Knoten sind von t aus über IJ Kanten erreichbar?

Beweis

- Klar nach (1)-(3) ❑

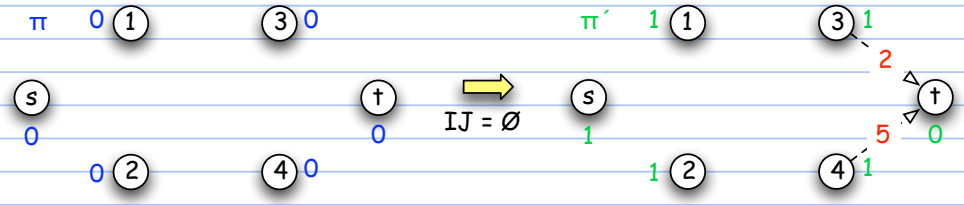
6.5 Beispiel

- Input Daten



$c_{ij} \geq 0 \Rightarrow \pi = 0$ ist zulässig in (D)

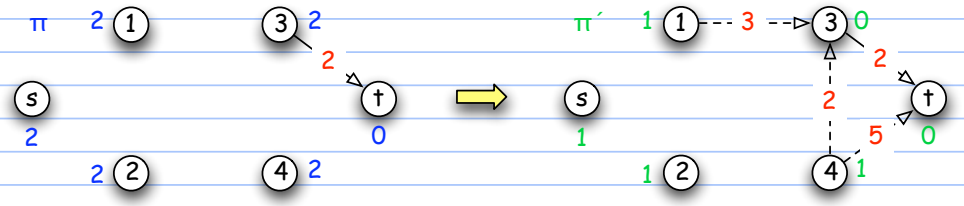
Iteration 1



$\theta_1 = \min\{ c_{ij} - (\pi_i - \pi_j) \mid (i,j) \notin IJ,\ \pi_i' - \pi_j' = 1 \} = 2$ für Kante (3,t)

$\Rightarrow \pi^* = \pi + \theta_1 \pi' = (0,\ldots,0)^T + 2 \cdot (1,1,1,1,1,0)^T = (2,2,2,2,2,0)^T$

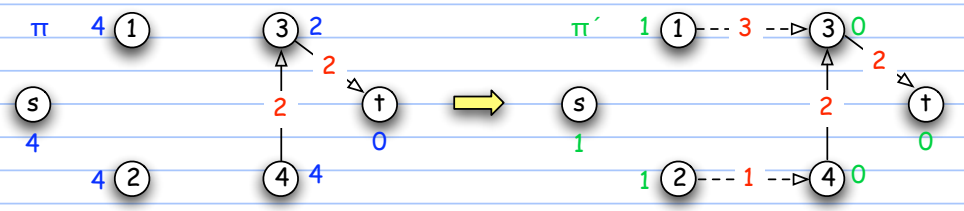
Iteration 2



$\theta_1 = \min\{ 3 - (2 - 2), 2 - (2 - 2), 5 - (2 - 0) \} = 2$ für Kante (4,3)

$\Rightarrow \pi^* = \pi + \theta_1 \pi' = (2,2,2,2,2,0)^T + 2 \cdot (1,1,1,0,1,0)^T = (4,4,4,2,4,0)^T$

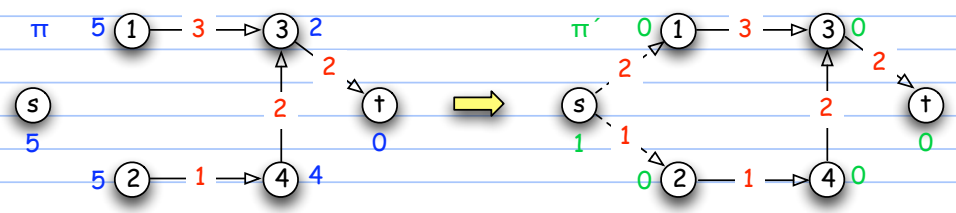
Iteration 3



$\theta_1 = \min\{ 3 - (4 - 2), 1 - (4 - 4) \} = 1$ für Kanten (1,3) und (2,4)

$\Rightarrow \pi^* = \pi + \theta_1 \pi' = (4,4,4,2,4,0)^T + 1 \cdot (1,1,1,0,0,0)^T = (5,5,5,2,4,0)^T$

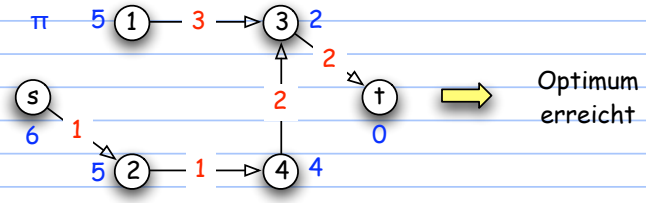
Iteration 4



$\theta_1 = \min\{ 2 - (5 - 5), 1 - (5 - 5) \} = 1$ für Kante (s,2)

$\Rightarrow \pi^* = \pi + \theta_1 \pi' = (5,5,5,2,4,0)^T + 1 \cdot (1,0,0,0,0,0)^T = (6,5,5,2,4,0)^T$

Iteration 5



$\pi = (6,5,5,2,4,0)^T$ ist dual optimal = kürzeste Entfernung bis t

Detaillierte Interpretation der einzelnen Schritte

(1) Definiere W als

$W := \{ i \in V \mid t \text{ ist von } i \text{ aus über Kanten aus IJ erreichbar} \} = \{ i \in V \mid \pi_i' = 0 \}$

π_i bleibt fest sobald $i \in W$ ist, da dann $\pi_i' = 0$ gilt

(2) Jede Kante (i, j), die zulässig wird, also in IJ kommt, bleibt in IJ,

denn dann ändern sich π_i und π_j um denselben Betrag $\Rightarrow \pi_i - \pi_j$ bleibt gleich

(3) $i \in W \Rightarrow \pi_i$ = Länge eines kürzesten Weges von i nach t

(induktiver Beweis)

In jeder Iteration des Algorithmus werden die Knoten aus V - W zu W hinzugefügt, die t am nächsten sind

(induktiver Beweis)

Folgerung

Der primal-duale Algorithmus für (SP) mit $c \geq 0$ ist im Wesentlichen der Dijkstra Algorithmus, wie beim Bindfadenmodell in Abschnitt 4.3

Herleitung der verschiedenen LPs (P), (D), (RP), (DRP)

Primales LP (P) und duales LP (D)

Wir betrachten die Formulierung des Transportproblem aus Abschnitt 4.1

(P) min $\sum_{i,j} c_{ij} f_{ij}$ unter

$\sum_j f_{ij} = a_i$ für alle i (alles wird in Knoten i abtransportiert)

$\sum_i f_{ij} = b_j$ für alle j (alles kommt in Knoten j an)

$f_{ij} \geq 0$ für alle i, j

mit (o.B.d.A.) $\sum_i a_i = \sum_j b_j$

Führe folgende Dual-Variable α_i, β_j ein

α_i $\sum_j f_{ij} = a_i$ für alle i

β_j $\sum_i f_{ij} = b_j$ für alle j

Dann lautet das duale LP

(D) max $\sum_i a_i\alpha_i + \sum_j b_j\beta_j$ unter

$\alpha_i + \beta_j \leq c_{ij}$ für alle i, j

α_i, β_j beliebig

Eine zulässige Ausgangslösung von (D) ist gegeben durch

$\alpha_i = 0$ für alle i

$\beta_j = \min_i c_{ij}$ für alle j (benötigt nicht $c_{ij} \geq 0$)

Restricted Primal (RP)

Die Menge der zulässigen Spalten ist

$IJ = \{ (i,j) \in E \mid \alpha_i + \beta_j = c_{ij} \}$

Als (RP) ergibt sich

min $\xi = \sum_{i=1,\dots,m+n} x_i^a$

$\sum_j f_{ij} + x_i^a = a_i$ für i = 1, ..., m

$\sum_i f_{ij} + x_{m+j}^a = b_j$ für j = 1, ..., n

$f_{ij} \geq 0$ für alle Kanten $(i,j) \in IJ$

$f_{ij} = 0$ für alle Kanten $(i,j) \notin IJ$

$x_i^a \geq 0$ für i = 1, ..., m+n

Wir modifizieren (RP) durch Substitution der Hilfsvariablen x_i^a in der Zielfunktion und erhalten (unter Berücksichtigung von $f_{ij} = 0$ für alle Kanten $(i,j) \notin IJ$)

$$\xi = \sum_i a_i + \sum_j b_j - 2 \sum_{(i,j) \in IJ} f_{ij}$$

($\sum_i a_i + \sum_j b_j$: konstant)

=> Minimierung von ξ <=> Maximierung von $\sum_{(i,j) \in IJ} f_{ij}$

Weglassen der künstlichen Variablen ergibt dann (wegen $x_i^a \geq 0$)

(RP') max $\sum_{(i,j) \in IJ} f_{ij}$

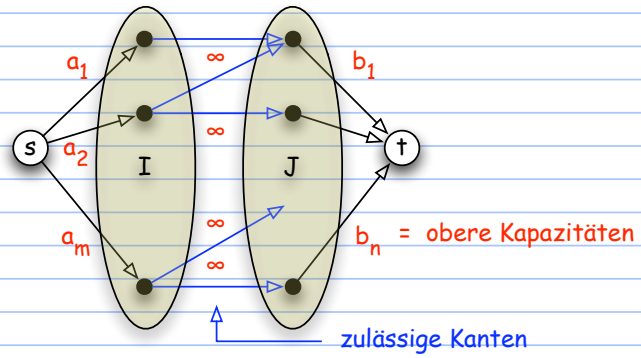
$\sum_j f_{ij} \leq a_i$ für $i = 1, \ldots, m$

$\sum_i f_{ij} \leq b_j$ für $j = 1, \ldots, n$

$f_{ij} \geq 0$ für alle Kanten $(i,j) \in IJ$

$f_{ij} = 0$ für alle Kanten $(i,j) \notin IJ$

=> (RP') entspricht einem Max-Fluss-Problem im Graphen G der zulässigen Kanten



Primal-dualer Algorithmus ergibt:

f ist optimal in (P) <=> der maximale Flusswert $v(f) = \sum_i a_i = \sum_j b_j$

Das duale (DRP) zu (RP)

Führe folgende Dual-Variablen u_i, v_j ein

u_i $\sum_j f_{ij} + x_i^a = a_i$ für $i = 1, \ldots, m$

v_j $\sum_i f_{ij} + x_{m+j}^a = b_j$ für $j = 1, \ldots, n$

Dann lautet das zu (RP) duale LP

(DRP) max $w = \sum_i a_i u_i + \sum_j b_j v_j$ unter

$u_i + v_j \leq 0$ für alle $(i,j) \in IJ$

$u_i,\ v_j \leq 1$

$u_i,\ v_j$ nicht vorzeichenbeschränkt

6.6 Lemma (Optimallösung von (DRP))

Sei $\xi_{opt} > 0$ in (RP) und sei f ein maximaler s,t-Fluss in G.

Sei $I^* \subseteq I$ die Menge der Knoten, zu denen in G_f ein Fluss erhöhender Weg existiert.

Sei $J^* \subseteq J$ die Menge der Knoten, zu denen in G_f ein Fluss erhöhender Weg existiert.

Dann ist durch

$\alpha_i' := 1$ falls $i \in I^*$ $\quad \alpha_i' := -1$ falls $i \notin I^*$

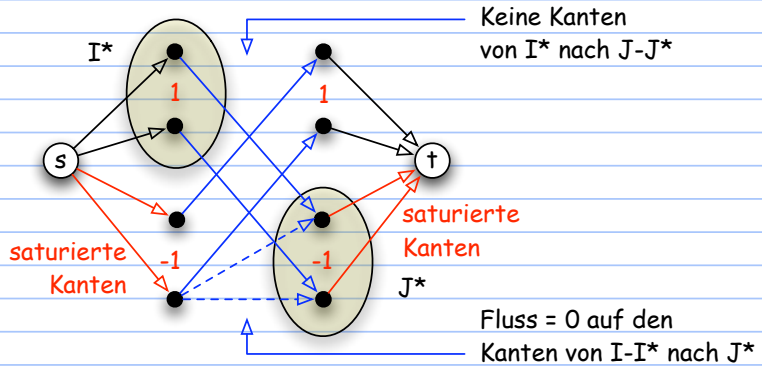
$\beta_j' := -1$ falls $j \in J^*$ $\quad \beta_j' := 1$ falls $j \notin J^*$

eine optimale Lösung von (DRP) gegeben.

Beweis

Aus ADM I ist bekannt, dass $X := \{s\} \cup I^* \cup J^*$ ein Schnitt minimaler Kapazität in G ist, der durch jeden Max-Fluss-Algorithmus ermittelt werden kann. Die Mengen I^* und J^* können also effizient berechnet werden.

Wir analysieren nun diesen Schnitt in G



(1) es gibt keine Kante (i,j) von I^* nach $J-J^*$

da andernfalls $j \in J^*$ wegen der unendlichen Kapazität von (i,j)

(2) $f_{ij} = 0$ für alle Kanten (i,j) von $I-I^*$ nach J^*

da andernfalls (j,i) eine Rückwärtskante in G_f wäre, was $i \in I^*$ ergäbe

(3) die Kanten (s,i) von s nach $I-I^*$ sind saturiert

da sonst $i \in I^*$ wäre

(4) die Kanten (j,t) von J^* nach t sind saturiert

da es sonst einen Fluss erhöhenden s,t-Weg gäbe

(5) der Flusswert ist $v(f) = \sum_{i \in I-I^*} a_i + \sum_{j \in J^*} b_j$

denn: $v(f)$ = Nettofluss von $X = \{s\} \cup I^* \cup J^*$ nach $V(G) - X$

=> Behauptung mit (1) - (4)

(A) α_i' und β_j' sind zulässig für (DRP)

α_i' und $\beta_j' \leq 1$ ist erfüllt

Annahme $\alpha_i' + \beta_j' > 0$ => $\alpha_i' = 1$ und $\beta_j' = 1$ => $i \in I^*$ und $j \in J-J^*$

=> Widerspruch zu (1)

(B) α_i' und β_j' sind optimal für (DRP)

Als Zielfunktionswert für α_i' und β_j' ergibt sich

$$w = \sum_i a_i\alpha_i' + \sum_j b_j\beta_j'$$

$$= \sum_{i \in I^*} a_i - \sum_{i \in I-I^*} a_i - \sum_{j \in J^*} b_j + \sum_{j \in J-J^*} b_i$$

Andererseits ist wegen (DRP') und (5)

$$\xi_{opt} = \sum_i a_i + \sum_j b_j - 2v(f)$$

$$= \sum_i a_i + \sum_j b_j - 2\left(\sum_{i \in I-I^*} a_i + \sum_{j \in J^*} b_j\right) = w$$

Dualitätssatz => α_i' und β_j' sind optimal ❑

Aktualisierung der dualen Lösung

Falls $\xi_{opt} > 0$, so gibt es 2 Fälle im primal-dualen Algorithmus (Satz 6.2)

Fall 1: $\alpha_i' + \beta_j' \leq 0$ für alle $(i,j) \notin IJ$

=> (P) unzulässig nach Satz 6.2

dies kann jedoch nicht auftreten, da (P) stets eine zulässige Lösung hat,

z.B. $f_{ij} = (1/\sum_k a_k) \cdot a_i \cdot b_j$

Fall 2: $\alpha_i' + \beta_j' > 0$ für ein $(i,j) \notin IJ$

Dieser Fall tritt also immer ein. Nach (6.7) ist

$$\theta_1 = \min\left\{ \frac{c_{ij} - \pi^T A_{ij}}{(\pi')^T A_{ij}} \mid (i,j) \notin IJ,\ (\pi')^T A_{ij} > 0 \right\}$$

$$= \min\left\{ \frac{c_{ij} - \alpha_i - \beta_j}{\alpha_i' + \beta_j'} \mid (i,j) \notin IJ,\ \alpha_i' + \beta_j' > 0 \right\}$$

$$= \min\left\{ \frac{c_{ij} - \alpha_i - \beta_j}{2} \mid i \in I^*,\ j \notin J^* \right\}$$

Wir fassen zusammen

6.7 Lemma (Aktualisierung der dualen Lösung)

Sei $\xi_{opt} > 0$ in (RP) und sei f ein maximaler s,t-Fluss in G.

Sei $I^* \subseteq I$ die Menge der Knoten, zu denen in G_f ein Fluss erhöhender Weg existiert.

Sei $J^* \subseteq J$ die Menge der Knoten, zu denen in G_f ein Fluss erhöhender Weg existiert.

Dann ist

$$\theta_1 = \min \{ \frac{c_{ij} - \alpha_i - \beta_j}{2} \mid i \in I^*, j \notin J^* \}$$

und die neue duale Lösung ergibt sich zu

$\alpha_i^* = \alpha_i + \theta_1$ falls $i \in I^*$ $\alpha_i^* = \alpha_i - \theta_1$ falls $i \notin I^*$

$\beta_j^* = \beta_j - \theta_1$ falls $j \in J^*$ $\beta_j^* = \beta_j + \theta_1$ falls $i \notin J^*$

Jeder optimale Fluss des alten (RP') bleibt zulässig im neuen (RP').

Beweis

Die neue duale Lösung ergibt sich allgemein als $\pi^* := \pi + \theta_1 \pi'$

Damit folgen die Werte von α_i^* und β_j^* aus dem Wert von θ_1 und Lemma 6.6

Zu zeigen bleibt, dass der optimale Fluss zulässig bleibt. Dies folgt bereits aus Satz 6.3, da der optimale Fluss eine Basislösung von (RP') ist, und wird hier noch einmal durch folgenden Claim verifiziert.

Claim: Kanten (i,j) mit positiven Fluss bleiben zulässig im neuen (RP)

$f_{ij} > 0$ => Kante (i,j) ist zulässig im alten (RP) => $\alpha_i + \beta_j = c_{ij}$

Analyse des Schnittes in G (Beweis Lemma 6.6) => zwei mögliche Fälle

Fall 1: $i \in I^*$ und $j \in J^*$

=> $\alpha_i^* + \beta_j^* = \alpha_i + \theta_1 + \beta_j - \theta_1 = \alpha_i + \beta_j = c_{ij}$

Fall 2: $i \in I - I^*$ und $j \in J - J^*$

$\alpha_i^* + \beta_j^* = \alpha_i - \theta_1 + \beta_j + \theta_1 = \alpha_i + \beta_j = c_{ij}$ ❑

Der primal-duale Algorithmus für das Transportproblem

Algorithmus alpha-beta

Input

Instanz des Transportproblem, d.h. Zahlen $a_i > 0$, $b_j > 0$, und c_{ij} mit $\sum_i a_i = \sum_j b_j$

Output

kostenminimale Transportlösung f_{ij}

Methode

bestimme eine zulässige Ausgangslösung von (D) durch

$\alpha_i = 0$ für alle i

$\beta_j = \min_i c_{ij}$ für alle j

repeat

bestimme den Graphen G der zulässigen Kanten aus $IJ := \{ (i,j) \in E \mid \alpha_i + \beta_j = c_{ij} \}$

berechne einen maximalen s,t-Fluss f in G // Warmstart vom Fluss der vorigen Iteration ist möglich

if $v(f) < \sum_i a_i$ then

setze $I^* := \{ i \in I \mid$ es gibt in G_f einen Fluss erhöhenden s,i-Weg $\}$

setze $J^* := \{ j \in J \mid$ es gibt in G_f einen Fluss erhöhenden s,j-Weg $\}$

setze

$$\theta_1 := \min \{ \frac{c_{ij} - a_i - \beta_j}{2} \mid i \in I^*, j \notin J^* \}$$

und als neue duale Lösung setze

$\alpha_i := \alpha_i + \theta_1$ falls $i \in I^*$ $\alpha_i := \alpha_i - \theta_1$ falls $i \notin I^*$

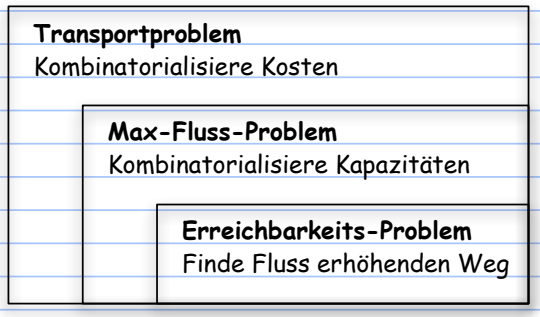
$\beta_j := \beta_j - \theta_1$ falls $j \in J^*$ $\beta_j := \beta_j + \theta_1$ falls $i \notin J^*$

until $v(f) = \sum_i a_i$

return Fluss f

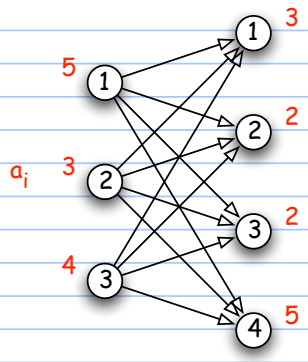
Interpretation

Das Paradigma des primal-dualen Algorithmus führt für das Transportproblem zu zwei inenandergeschachtelten Schleifen von Erreichbarkeitsproblemen. In jeder Schleife werden die Kosten "kombinatorialisiert".



6.8 Beispiel

Input Daten



a_i b_j

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

die c_{ij} Matrix

Bestimmung der dualen Ausgangslösung

β_j 1 1 2 2

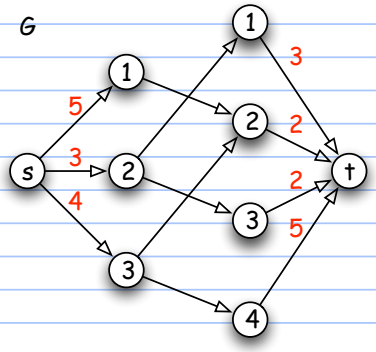
α_i		1	2	3	4
0	1	2	(1)	3	4
0	2	(1)	2	(2)	3
0	3	3	(1)	4	(2)

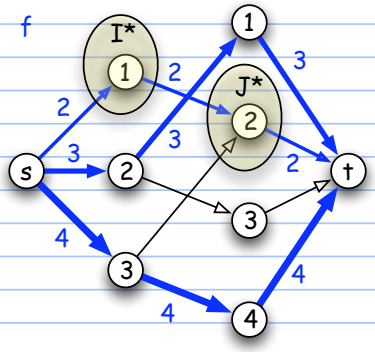
(○) = $\min_i c_{ij}$
entspricht den (i,j)
die zulässige Kanten sind

Iteration 1

Bestimmung des Graphen G der zulässigen Kanten und Berechnung eines maximalen s,t-Flusses in G und der

Mengen I* und J*





Flusswert $v(f) = 9 < \sum_i a_i = 12$ => duale Lösung aktualisieren

Aktualisierung der dualen Lösung

Berechnung von θ_1

$$\theta_1 := \min \{ \frac{c_{ij} - a_i - \beta_j}{2} \mid i \in I^*, j \notin J^* \}$$

	β_j	1	1	2	2
α_i		1	2	3	4
0	1	2	1	3	4
0	2	1	2	2	3
0	3	3	1	4	2

c_{ij} Matrix

$I^* = \{ 1 \}, \; J^* = \{ 2 \}$

Kante (1,1) -> (2 - 0 - 1) / 2 = 1/2

Kante (1,3) -> (3 - 0 - 2) / 2 = 1/2 => $\theta_1 = 1/2$

Kante (1,4) -> (4 - 0 - 2) / 2 = 1

Berechnung der neuen dualen Lösung

$\alpha_i := \alpha_i + \theta_1$ falls $i \in I^*$ $\alpha_i := \alpha_i - \theta_1$ falls $i \notin I^*$

$\beta_j := \beta_j - \theta_1$ falls $j \in J^*$ $\beta_j := \beta_j + \theta_1$ falls $i \notin J^*$

Also $\alpha_1 = 1/2$ $\alpha_2 = -1/2$ $\alpha_3 = -1/2$

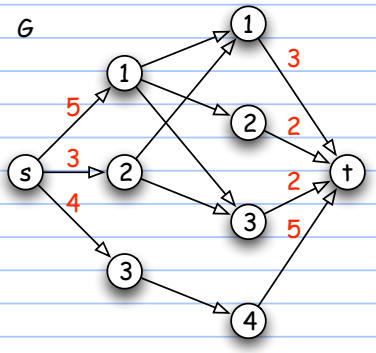
$\beta_1 = 3/2$ $\beta_2 = 1/2$ $\beta_3 = 5/2$ $\beta_4 = 5/2$

Iteration 2

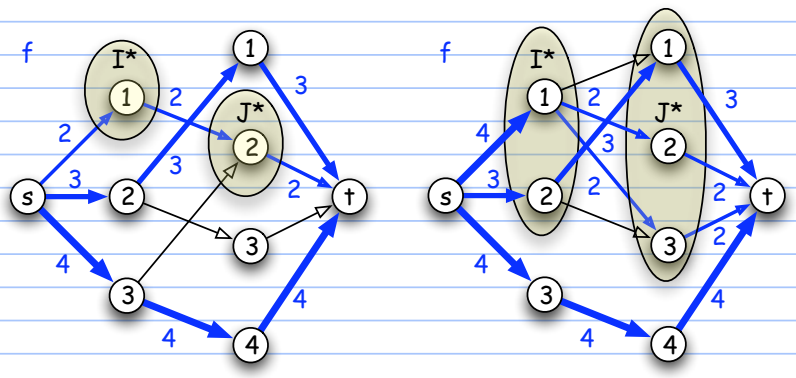
Bestimmung des Graphen G der zulässigen Kanten

	β_j	$\frac{3}{2}$	$\frac{1}{2}$	$\frac{5}{2}$	$\frac{5}{2}$
α_i		1	2	3	4
1/2	1	2	1	3	4
-1/2	2	1	2	2	3
-1/2	3	3	1	4	2

entspricht zulässigen Kanten



Berechnung eines maximalen s,t-Flusses in G und der Mengen I* und J*



bisheriger Fluss neuer Fluss

Flusswert $v(f) = 11 < \sum_i a_i = 12$ => duale Lösung aktualisieren

Aktualisierung der dualen Lösung

Berechnung von θ_1

$$\theta_1 := \min \{ \frac{c_{ij} - \alpha_i - \beta_j}{2} \mid i \in I^*, j \notin J^* \}$$

	β_j	3/2	1/2	5/2	5/2
α_i		1	2	3	4
1/2	1	2	1	3	4
-1/2	2	1	2	2	3
-1/2	3	3	1	4	2

$I^* = \{ 1, 2 \}$, $J^* = \{ 1, 2, 3 \}$

Kante (1,4) -> (4 - 1/2 - 5/2) / 2 = 1/2

Kante (2,4) -> (3 + 1/2 - 5/2) / 2 = 1/2 => $\theta_1 = 1/2$

Berechnung der neuen duale Lösung

$\alpha_i := \alpha_i + \theta_1$ falls $i \in I^*$ $\alpha_i := \alpha_i - \theta_1$ falls $i \notin I^*$

$\beta_j := \beta_j - \theta_1$ falls $j \in J^*$ $\beta_j := \beta_j + \theta_1$ falls $i \notin J^*$

Also $\alpha_1 = 1$ $\alpha_2 = 0$ $\alpha_3 = -1$

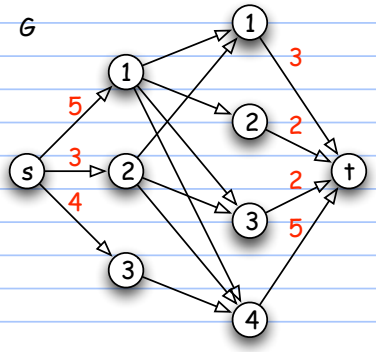
$\beta_1 = 1$ $\beta_2 = 0$ $\beta_3 = 2$ $\beta_4 = 3$

Iteration 3

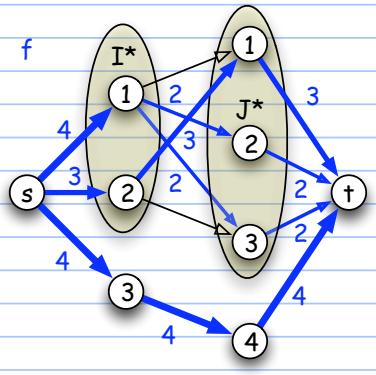
Bestimmung des Graphen G der zulässigen Kanten

α_i \ β_j	1	2	3	4
	1	0	2	3
1	2	1	3	4
0	1	2	2	3
-1	3	1	4	2

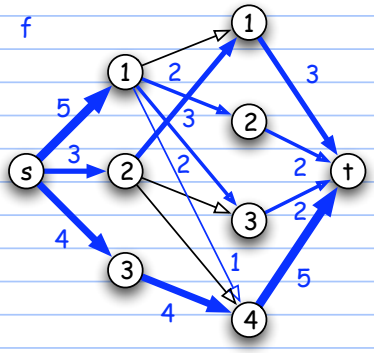
○ entspricht zulässigen Kanten



Berechnung eines maximalen s,t-Flusses in G und der Mengen I* und J*



bisheriger Fluss



neuer Fluss

Flusswert $v(f) = 12 = \sum_i a_i = 12$ => optimale Lösung gefunden

Aufwand des Algorithmus

Sei o.B.d.A. $m \leq n$ => G hat $O(n)$ Knoten und $O(n^2)$ Kanten

Bei jeder Ermittlung eines Fluss erhöhenden Weges erhöht sich der primale Zielfunktionswert, und dieser ist durch $\sum_i a_i$ beschränkt.

=> Gesamtaufwand für Flusserhöhung ist $(\Sigma_i a_i) \cdot O(\text{Breitensuche}) = (\Sigma_i a_i) O(n^2)$

Alle anderen Berechnungen (Ermittlung von G, θ_1, neue duale Lösung) sind in $O(n^2)$ und geschehen maximal $(\Sigma_i a_i)$ oft

=> Gesamtaufwand des primal-dualen Algorithmus ist $(\Sigma_i a_i) O(n^2)$

=> der primal-duale Algorithmus ist nur pseudo-polynomial

Eine Verbesserung ergibt sich durch Capacity Scaling (vgl. ADM I, Abschnitt 6.4) bei den a_i und b_j

=> im Wesentlichen nur $\log(\max\{a_i, b_j\})$ viele Flussprobleme mit Laufzeit $O(n^3)$

Interessant ist der Spezialfall $a_i = b_j = 1$ => $n = m$ (Zuordnungsproblem, Assignment Problem)

=> $\Sigma_i a_i = n$ => $O(n^3)$ insgesamt

Dieser Zugang wurde zuerst für das Gewichtete Maximale Matching Problem in bipartiten Graphen entwickelt (Paul Kuhn 1955) und ist als Ungarische Methode bekannt, vgl. z.B.

A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency

Band 1, Kapitel 17.2

Ziel dieses Abschnitts

Skizze eines primal dualen Algorithmus für gewichtete (perfekte) Matchings.

Dies schließt die Lücke aus ADM I, Abschnitt 7.1

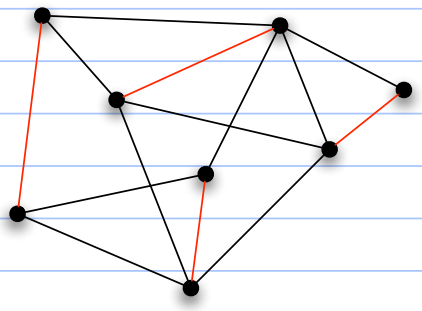
Matchings

Sei G ein ungerichteter Graph. Ein Matching in G ist

eine Menge $M \subseteq E(G)$ von Kanten so dass je 2 Kanten aus M keinen Endpunkt gemeinsam haben

ein Matching M heißt perfekt falls jeder Knoten in G mit einer Kante aus M inzidiert

ein Graph mit einem perfekten Matching (rote Kanten)



- Maximum Weight Matching Problem (MWMP)
  - Instanz
    - Ungerichteter Graph G, Kantengewichte c(e)
  - Aufgabe
    - Finde ein Matching M mit maximalem Gewicht c(M)

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

- Minimum Weight Perfect Matching Problem (MWPMP)
  - Instanz
    - Ungerichteter Graph G, Kantengewichte c(e)
  - Aufgabe
    - Finde ein perfektes Matching M mit minimalem Gewicht c(M)

      oder stelle fest, dass kein perfektes Matching existiert

- 6.8 Lemma (Äquivalenz von Matching) Problemen

- MWMP und MWPMP sind äquivalent in dem Sinne, dass eine einfache Transformation die Probleme in einander überführt, so dass aus der optimalen Lösung des einen eine optimale Lösung des anderen konstruiert werden kann.
  - Beweis
    - "=>"
      - Sei (G,c) eine Instanz des Minimum Weight Perfect Matching Problem.

        Wähle K so groß, dass c'(e) := K - c(e) > 0 für alle Kanten e und nur ein kardinalitäts-maximales Matching von G maximales Gewicht bzgl. c' hat. (K := 1 + $\sum_{e \in M} |c(e)|$ tut es)
      - Sei M eine optimale Lösung des Maximum Weight Matching Problem für (G, c')

        M kardinalitäts-maximal => M ist perfektes Matching für (G,c) oder es gibt kein perfektes Matching in G
      - Falls M perfekt, so ist c'(M) = Kn/2 - $\sum_{e \in M} c(e)$. M hat also genau dann maximales Gewicht bzgl. c' wenn M minimales Gewicht bzgl. c hat.
    - "<="
      - Sei (G,c) eine Instanz des Maximum Weight Matching Problem

        Füge |V(G)| viele neue Knoten zu G hinzu und soviele Kanten, dass der neue Graph G' vollständig wird.

Setze $c'(e) := -c(e)$ falls $e \in E(G)$ und $c'(e) := 0$ falls e eine neue Kante ist.

Sei M' eine optimale Lösung des Minimum Weight Perfect Matching Problem für (G', c')

=> $M := M' \cap E(G)$ ist eine optimale Lösung des Maximum Weight Matching Problem ❏

Der primal-duale Algorithmus für das Minimum Weight Perfect Matching Problem

Primales LP (P)

Es ist keineswegs klar, wie eine LP-Formulierung aussehen soll. Der folgende Satz war eine der großen Leistungen von Edmonds. Er betrachtet zu einer Instanz (G,c) mit $G = (V, E)$ das LP (P)

$\min \sum_{e \in E} c(e)x_e$

$x(\delta(v)) = 1$ für alle $v \in V$

$x(\delta(S)) \geq 1$ für alle ungeraden Knotenmengen S von G

$x \geq 0$

Dabei ist $x(\delta(S)) := \sum_{e \in \delta(S)} x_e$

6.9 Satz (Matching Polytop Theorem, Edmonds 1965)

Sei Sei (G,c) eine Instanz des Minimum Weight Perfect Matching Problem. Dann gilt

(1) G hat ein perfektes Matching <=> (P) hat eine zulässige Lösung

(2) In diesem Fall ist das Minimalgewicht eines perfekten Matchings von G gleich dem Optimalwert von (P).

Beweis

Der primal-duale Algorithmus konstruiert (im Falle der Lösbarkeit) eine optimale Lösung x von (P), die ein perfektes Matching von G ist.

=> Satz 6.9 und wegen Lemma 3.5, dass alle zulässigen Basislösungen von (P) zu perfekten Matchings korrespondieren. ❏

Duales LP (D)

Wir verzichten darauf, (P) in Standardform zu bringen. Der primal-duale Algorithmus gilt natürlich sinngemäß auch für andere Formen von (P).

Sei U die Menge aller ungeraden Knotenmengen von G. Das duale LP zu (P) ist (D)

$\max \Sigma(y_v : v \in V) + \Sigma(Y_S : S \in U)$

$y_v + y_w + \Sigma(Y_S : e \in S \in U) \leq c(e)$ für alle Kanten $e = vw \in E$

$Y_S \geq 0$ für alle $S \in U$

y_v nicht vorzeichenbeschränkt

- Bedingungen vom komplementären Schlupf
  - $x_e > 0 \Rightarrow c(e) - (y_v + y_w + \Sigma(Y_S : e \in S \in U)) = 0$
  - $Y_S > 0 \Rightarrow x(\delta(S)) = 1$
  - Ist x Inzidenzvektor eines perfekten Matching M so sind diese Bedingungen äquivalent zu

    $e \in M \Rightarrow c(e) - (y_v + y_w + \Sigma(Y_S : e \in S \in U)) = 0$ (6.8)

    $Y_S > 0 \Rightarrow |M \cap \delta(S)| = 1$ (6.9)

- Arbeitsweise des primal-dualen Algorithmus
  - (6.8) definiert "zulässige" Kanten für das gesuchte Matching
  - (6.9) entspricht $|M \cap \delta(v)| = 1$, wenn S zum Pseudoknoten v geschrumpft ist (Blüte)
  - => Lösung von (RP) entspricht im Wesentlichen Suche eines perfekten (also maximalen) Matchings im Graph der zulässigen Kanten, in dem alle Mengen S mit $Y_S > 0$ geschrumpft sind
  - => Lösung mit Algorithmus für Matching maximaler Kardinalität
  - Optimale Lösung von (DRP) kann wie beim Transportproblem direkt aus dem besten Matching von (RP) abgelesen werden, ist jedoch etwas komplizierter
  - Insgesamt wird das Minimum Weight Perfect Matching Problem auf eine Folge von Kardinalitäts-maximalen

Matching Problemen reduziert

- Der Algorithmus kann mit einer Laufzeit von $O(n^2m)$ implementiert werden. Insbesondere sind stets nur maximal n Variablen $Y_S > 0$.

  Ferner sind Verbesserungen für dicht besetzte Graphen möglich (arbeite auf dünnen Teilmengen der Kanten)
- Details siehe Kapitel 5.3 in

  W. J. Cook, W. H. Cunningham, W. R. Pulleyblank und A. Schrijver

  Combinatorial Optimization

  Wiley 1998

7.1 Einführung

- Ganzzahlige lineare Programme (Integer Linear Program, ILP, IP) verlangen die Ganzzahligkeit der Variablen für ...
- In diesem Abschnitt:
  - Die Ganzzahligkeit erhöht die Modellierungskraft enorm, viele nichtlineare Effekte können so modelliert werden
  - Daher erhält man i.A. NP-schwere Probleme

- LP Relaxation eines IP
  - Standardform eines IP

    $\min\ c^T x$

    unter $Ax = b$

    $x \geq 0$ und ganzzahlig

    Spezialfall: 0/1 IP oder Binary Integer Program

    $\min\ c^T x$

    unter $Ax = b$

    $x_j \in \{0, 1\}$
  - Die LP Relaxation eines IP ergibt sich durch Weglassen der Ganzzahligkeitsbedingungen, d.h.

    $\min\ c^T x$

unter $Ax = b$

$x \geq 0$

im allgemeinen Fall und

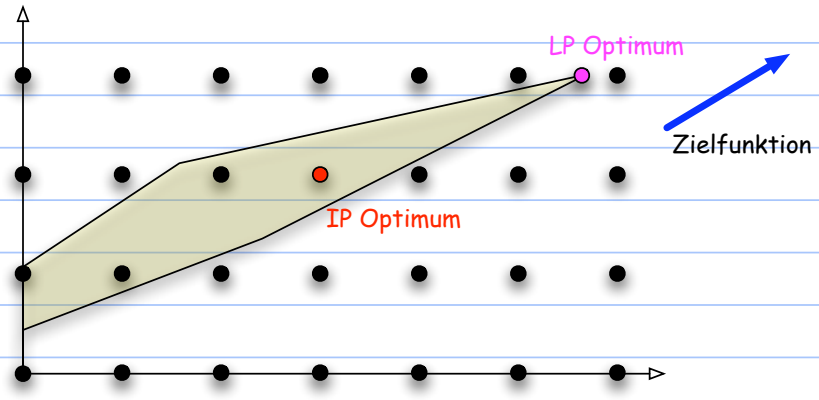
$\min\ c^T x$

unter $Ax = b$

$0 \leq x_j \leq 1$

im 0/1 Fall

- Lösen der LP Relaxation und anschließendes Runden der Variablen
  - erzeugt i.A. keine zulässige Lösung
  - Ohne weitere Überlegungen nur sinnvoll bei großen Werten der Variablen, aber auch dann sind große Fehler möglich



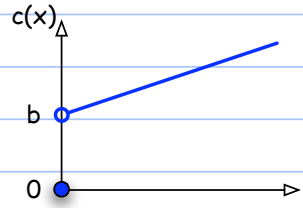
- Sind die $x_j \in \{0, 1\}$ (Entscheidungsvariable)

  z.B. die f_{ij} beim Kürzeste-Wege-Problem

  so ist Runden einer nicht ganzzahligen Lösung zunächst einmal nicht sinnvoll

  (später vorsichtig genutzt für Approximationsalgorithmen)

- Modellierungsmöglichkeiten mit IPs
  - (1) Fixkosten (Fixed Charge Problem)



$$c(x) = \begin{cases} ax + b & \text{falls } x > 0 \\ 0 & \text{falls } x = 0 \end{cases}$$

$x \in \mathbb{R}^1$, $a, b > 0$

- Führe 0/1-Variable δ und die Restriktion $x \leq \delta \cdot U$ ein,
  wobei U eine obere Schranke für die Werte von x ist
- Claim: Kostenfunktion ist jetzt modellierbar als $c(x,\delta) = ax + b\delta$
  - $x > 0 \Rightarrow \delta = 1 \Rightarrow c(x) = ax + b$
    $x = 0 \Rightarrow \delta = 0$ im Optimum, da $a, b > 0 \Rightarrow c(x) = 0$ ❑

- (2) Disjunktive Bedingungen
  - $x \geq a$ oder $y \geq b$ mit $a, b \geq 0$ und $x, y \geq 0$
  - Führe 0/1-Variable δ ein
  - Claim: Die Ungleichungen $x \geq \delta a$ und $y \geq (1-\delta)b$ modellieren die disjunktive Bedingung
    - klar, da δ eine 0/1 Variable ❑

- (3) Konditionale Bedingungen
  - if $x < a$ then $y \geq b$ else $y \geq 0$ mit $a, b > 0$
  - Claim: Die Konditionale Bedingung kann auf Fall (2) reduziert werden
    - die Konditionale Bedingung ist äquivalent zu

      $y \geq 0$

      $x \geq a$ oder $y \geq b$ ❑

- (4) Diskrete Variable
  - $x \in \{ s_1, \ldots, s_m \}$
  - Claim: Diskrete Variable $x \in \{ s_1, \ldots, s_m \}$ können modelliert werden durch

    $x = s_1\delta_1 + \ldots + s_m\delta_m$ mit $\delta_j \in \{ 0, 1 \}$ und $\delta_1 + \ldots + \delta_m = 1$
    - klar ❑

- 7.1 Beispiel (Minimum Weight Perfect Matching Problem als IP)
  - Jede Lösung des IP

$\min \sum_{e \in E} c(e)x_e$

$x(\delta(v)) = 1$ für alle $v \in V$

$x_e \in \{0, 1\}$

ist ein perfektes Matching

Komplexität von ILPs

7.2 Satz (Komplexität von ILPs)

(1) SATISFIABILITY (SAT) ist reduzierbar auf ILP

(2) Es ist NP-schwer zu entscheiden, ob ein ILP eine zulässige Lösung hat

(3) Es ist NP-schwer, von einer zulässigen Lösung der LP Relaxation eines ILP auf eine zulässige Lösung des ILP zu runden

Beweis

Sei eine Instanz von SAT gegeben durch m Klauseln $C_1, \ldots, C_m$ in Booleschen Variablen $x_1, \ldots, x_n$

Führe für jede Boolesche Variable x_i eine 0/1-Variable z_i ein mit $z_i = 1$ falls x_i = TRUE

Dann lässt sich die Erfüllung einer Klausel als lineare Ungleichung schreiben und die Existenz einer erfüllenden Belegung ist äquivalent zur Existenz einer zulässigen Lösung für das ILP.

Beispiel:

$$\underbrace{x_1 \vee x_2 \vee x_3}_{C_1}, \underbrace{x_1 \vee \bar{x}_2}_{C_2}, \underbrace{x_2 \vee \bar{x}_3}_{C_3}, \underbrace{x_3 \vee \bar{x}_1}_{C_4}, \underbrace{\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3}_{C_5}$$

ist äquivalent zu

$z_1 + z_2 + z_3 \geq 1$

$z_1 + (1-z_2) \geq 1$

$z_2 + (1-z_3) \geq 1$

$z_3 + (1-z_1) \geq 1$

$(1-z_1) + (1-z_2) + (1-z_2) \geq 1$

$z_i \in \{0, 1\}$

Hat jede Klausel ≥ 2 Literale (dies ist der nicht-triviale Fall), so ist $z_i = 1/2$ eine zulässige Lösung der LP Relaxation. Das Runden auf eine zulässige Lösung des ILP ist daher genauso schwer wie das Finden einer efüllenden Belegung für die gegebene SAT Instanz. ❑

Beachte: Der Beweis zeigt nicht, dass der Test auf Zulässigkeit NP-vollständig ist. Dazu müssten wir zeigen, dass ein Zertifikat für Zulässigkeit von polynomialer Länge existiert (vgl. ADM I). Das ist zunächst unklar, es kann jedoch gezeigt werden, dass die Komponenten x_i einer ganzzahligen zulässigen Lösung x nicht zu groß

werden (eine analoge Aussage zu Lemma 3.4). Daher kann x selbst als Zertifikat genommen werden. NP-schwer kann also überall in Satz 7.2 durch NP-vollständig ersetzt werden.

Fragestellung dieses Abschnittes

Wann hat ein LP ganzzahlige Basislösungen?

=> Dann lässt sich ein ILP dadurch lösen indem man die LP Relaxation mit dem Simplexalgorithmus löst.

Hier der Speziallfall:

Wann hat $Ax = b$ nur ganzzahlige Basislösungen für beliebige Wahl von gannzzahligen rechten Seiten b?

Dies ist dann eine Eigenschaft der Matrix A

Vollständig unimodulare Matrizen

Eine quadratische Matrix B mit ganzzahligen Einträgen heißt unimodular

:<=> $\det B \in \{ -1, 1 \}$

Eine Matrix A mit ganzzahligen Einträgen heißt vollständig unimodular (totally unimodular, TUM)

:<=> jede quadratische nicht-singuläre Teilmatrix ist unimodular

Erste Eigenschaften

- A TUM => A hat nur Einträge $a_{ij} \in \{ -1, 0, 1 \}$
- die kleinste nicht unimodulare Matrix ist

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

Ist B eine Basis von A mit $B = (A_{B(1)}, A_{B(2)}, \ldots, A_{B(m)})$, so folgt aus der Cramerschen Regel

$$x_{B(i)} = \frac{\det B^i}{\det B} \quad \text{mit } B^i = (A_{B(1)}, \ldots, A_{B(i-1)}, b, A_{B(i+1)}, \ldots, A_{B(m)})$$

=> $x_{B(i)}$ ganzzahlig falls A TUM und b ganzzahlig

Polyeder von linearen Optimierungsproblemen mit ganzzahligen Ecken

Sei $R_1(A) := \{ x \in \mathbb{R}^n \mid Ax = b, x \geq 0 \}$ das Polyeder zur Standardform des LP

Sei $R_2(A) := \{ x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0 \}$ das Polyeder zur kanonischen Form des LP

Bemerkung:

Beide Polyeder sind hier als Teilmengen des $\mathbb{R}^n$ definiert.

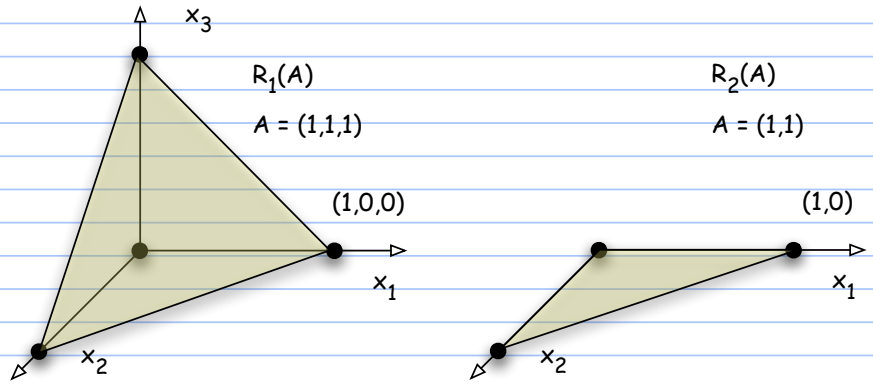
Die Definition $R_2(A)$ entspricht der in Abschnitt 3.3 betrachtenen Korrespondenz zwischen geometrischer und algebraischer Interpretation von LPs, insbesondere entsprechen die Ecken von $R_2(A)$ den zulässigen Basislösungen des um Schlupfvariablen erweiterten LP $\{ Ax + s = b, x, s \geq 0 \}$

Die Ecken von $R_1(A)$ entsprechen ebenfalls den zulässigen Basislösungen des LP $\{ Ax = b, x \geq 0 \}$ denn:

Ist x zulässige Basislösung zur Basis B, so ist $x_N = 0$, d.h. x liegt im Schnitt der n-m Hyperebenen $x_i = 0$ mit $i \in N$ und der m Hyperebenen $a_i x = b_i$, $i = 1, \ldots, m$.

Die Umkehrung folgt mit ähnlichen Überlegungen zu denen im Beweis zu Satz 3.6.

Beispiel:



Hier hat x_3 die Rolle der Schlupfvariablen für $R_2(A)$

7.3 Satz (Ganzzahligkeit von $R_1(A)$)

Ist A vollständig unimodular, so sind alle Ecken von $R_1(A)$ für beliebige ganzzahlige rechte Seite b ganzzahlig.

Insbesondere führt der Simplexalgorithmus in einem LP in Standardform mit vollständig unimodularer Matrix A bei ganzzahliger rechter Seite b immer zu einer ganzzahligen Optimallösung.

Beweis:

folgt aus dem Abschnitt "Erste Eigenschaften" ❑

7.4 Satz (Ganzzahligkeit von $R_2(A)$)

Ist A vollständig unimodular, so sind alle Ecken von $R_2(A)$ für beliebige ganzzahlige rechte Seite b ganzzahlig.

Insbesondere führt der Simplexalgorithmus in einem LP in kanonischer Form mit vollständig unimodularer Matrix A bei ganzzahliger rechter Seite b nach Einführung von Schlupfvariablen immer zu einer ganzzahligen Optimallösung.

Beweis:

Nach Einführung von Schlupfvariablen entsteht die Matrix $(A|I)$.

Sei C eine nicht-singuläre quadratische Teilmatrix von $(A|I)$

=> nach geeigneter Permutation der Zeilen hat C die Form

$$\left(\begin{array}{c|c} B & 0 \\ \hline D & I_k \end{array}\right)$$

mit B = quadratische Teilmatrix von A

I_k = (k,k)-Einkeitsmatrix

=> $|\det(C)| = |\det(B)| = 1$, da A TUM

=> $(A|I)$ ist TUM

=> Behauptung mit Satz 7.3 und Satz 3.10 ❑

Die Sätze 7.3 und 7.4 bedeuten also, dass die Polyeder $R_1(A)$ und $R_2(A)$ ganzzahlige Ecken haben, wenn A vollständig unimodular und die rechte Seite b ganzzahlig ist.

Erkennung von vollständig unimodularen Matrizen

Die Komplexität der Erkennung vollständig unimodularer Matrizen war lange offen und wurde erst durch

Seymour 1980 gelöst, der in einem "Dekompositionssatz" zeigte, dass sich jede vollständig unimodulare Matrix aus "einfachen" vollständig unimodularen Matrizen nach gewissen Konstruktionsprinzipien ergibt. Hieraus lässt sich ein polynomialer Algorithmus zur Erkennung vollständig unimodularer Matrizen herleiten mit einer Laufzeit von $O((m+n)^4 m)$

Details siehe Kapitel 19 und 20 in

A. Schrijver

Theory of Linear and Integer Programming

Wiley 1986

Hier nur ein hinreichendes Kriterium

7.5 Satz (Ein hinreichendes Kriterium für vollständige Unimodularität)

Eine Matrix A mit Einträgen $a_{ij} \in \{ -1, 0, 1 \}$ ist vollständig unimodular, falls sie folgende Bedingungen erfüllt

(1) A hat pro Spalte maximal 2 Einträge $\neq 0$

(2) Die Zeilen von A können in zwei disjunkte Mengen I_1, I_2 aufgeteilt werden mit

Für jede Spalte mit 2 Einträgen $\neq 0$ und gleichem Vorzeichen liegen die zugehörigen Zeilen in

verschiedenen I_j

Für jede Spalte mit 2 Einträgen $\neq 0$ und verschiedenen Vorzeichen liegen die zugehörigen Zeilen in demselben I_j

Beweis durch Induktion nach der Größe k der quadratischen Teilmatrix

Induktionsanfang k = 1

klar, da A nur Einträge $a_{ij} \in \{ -1, 0, 1 \}$ hat

Schluss auf k

Sei C eine quadratische nichtsinguläre (k,k)-Teilmatrix von A

=> in jeder Spalte von C gibt es mindestens einen Eintrag

Fall 1: es gibt in C eine Spalte mit genau einem Eintrag $a_{ij} \neq 0$

entwickle $\det(C)$ nach dieser Spalte, wobei C' die Untermatrix von C nach Streichen von Zeile i und Spalte j ist

=> $|\det(C)| = |a_{ij}| \cdot |\det(C')|$

C nicht singulär => $|\det(C')| \neq 0$

Induktionsvoraussetzung => $|\det(C')| = 1$

$a_{ij} \in \{ -1, 1 \}$ => $|\det(C)| = 1$

Fall 2: alle Spalten von C haben mindestens 2 Einträge $\neq 0$

(1) => alle Spalten haben genau 2 Einträge $\neq 0$

Betrachte die Aufteilung der Zeilen in I_1, I_2 gemäß (2)

=> für jede Spalte j ist $\sum_{i \in I_1} a_{ij} = \sum_{i \in I_2} a_{ij}$

=> $\sum_{i \in I_1} a_i - \sum_{i \in I_2} a_i = 0$

d.h. eine Linearkombination der Zeilenvektoren von C ergibt den Nullvektor

=> Widerspruch zu C nicht-singulär

=> dieser Fall kann nicht auftreten ❑

7.6 Korollar (Wichtige vollständig unimodulare Matrizen)

Jedes LP in Standardform oder kanonischer Form, dessen Koeffizientenmatrix gleich der

1. Knoten-Kanten Inzidenzmatrix eines Digraphen
2. Knoten-Kanten Inzidenzmatrix eines bipartiten Graphen

ist, hat nur ganzzahlige optimale Basislösungen (bei ganzzahliger rechter Seite b).

Hierzu gehören u.A. die LP Formulierungen des

Kürzeste-Wege-Problem

Max-Fluss-Problem

Transportproblem

Beweis

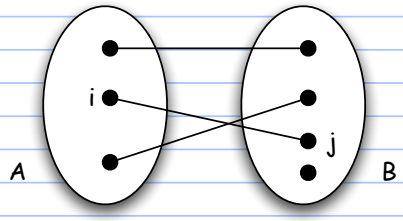
Fall 1

A enthält in diesem Fall pro Spalte genau eine +1 und eine -1

=> setze I_1 = Menge aller Zeilen, $I_2 = \emptyset$

Fall 2

Sei G der bipartite Graph mit Bipartition A und B



=> die Spalte zur Kante ij enthält genau 2 Einträge $\neq 0$, und zwar eine +1 für den Knoten i und eine +1 für den Knoten j

=> setze $I_1 = A$, $I_2 = B$ ❑

Der Satz von Birkhoff & von Neumann über doppelt stochastische Matrizen

Wir beweisen mit unseren Mitteln einen bekannten und nützlichen Satz über doppelt stochastische Matrizen

Eine nxn-Matrix mit Einträgen $0 \le a_{ij} \le 1$ heißt doppelt stochastisch

:<=> jede Zeilensumme und jede Spaltensumme ergibt 1

Eine nxn-Matrix mit Einträgen $a_{ij} \in \{ 0, 1 \}$ heißt Permutationsmatrix

:<=> in jeder Zeile und in jeder Spalte steht genau eine 1

7.7 Satz (Birkhoff 1946, von Neumann 1953)

Jede doppelt stochastische nxn-Matrix lässt sich als Konvexkombination von nxn-Permutationsmatrizen darstellen

Beweis

Eine doppelt stochastische Matrix M lässt sich auffassen als zulässige Lösung des Zuordnungsproblems

$\min \sum_{i,j} c_{ij} f_{ij}$ unter

$\sum_j f_{ij} = 1$ für alle $i = 1, \ldots, n$

$\sum_i f_{ij} = 1$ für alle $j = 1, \ldots, n$

$f_{ij} \ge 0$ für alle i, j

Sei A die zugehörige Koeffizientenmatrix und $R_1(A)$ das zugehörige Polyeder zur Standardform

$R_1(A)$ ist ein Polytop, da der Zulässigkeitsbereich wegen $0 \le f_{ij} \le 1$ beschränkt ist

Satz von Minkowski (Satz 3.9) => M ist Konvexkombination der Ecken von $R_1(A)$

A ist dabei Knoten-Kanten Inzidenzmatrix des vollständigen bipartiten Graphen $K_{n,n}$

=> A ist vollständig unimodular nach Korollar 7.6

=> Die Ecken von $R_1(A)$ sind ganzzahlig nach Satz 7.3

$0 \le f_{ij} \le 1$ => die Ecken von $R_1(A)$ sind Permutationsmatrizen ❑

Ziel dieses Abschnitts

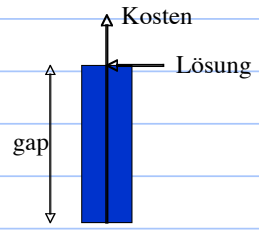
Vorstellung von Branch and Bound als eine Standard-Technik zur exakten Lösung NP-vollständiger Probleme, speziell von IPs.

Obwohl einfach, ist Branch and Bound die Grundlage und das Arbeitspferd für alle kommerziellen Codes zur Lösung von IPs, allerdings angereichert mit einer Vielzahl von Verbesserungen und Tricks.

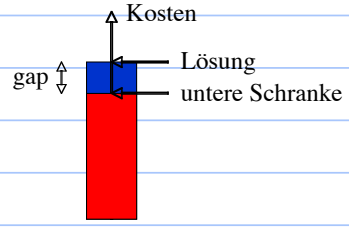
Grundidee von Branch and Bound

Branch and Bound (B&B) = geschickt (Problem-abhängig) organisierte systematische Durchforstung der Menge der zulässigen Lösungen nach einer Optimallösung oder bis zum Abbruch mit einer guten Lösung (d.h. mit einer Instanz-abhängigen Gütegarantie)

Der Nutzen unterer Schranken bei der Minimierung



wissen nicht wie gut eine zulässige Lösung ist, wenn sie heuristisch erzeugt wird (z.B. mit lokaler Suche)



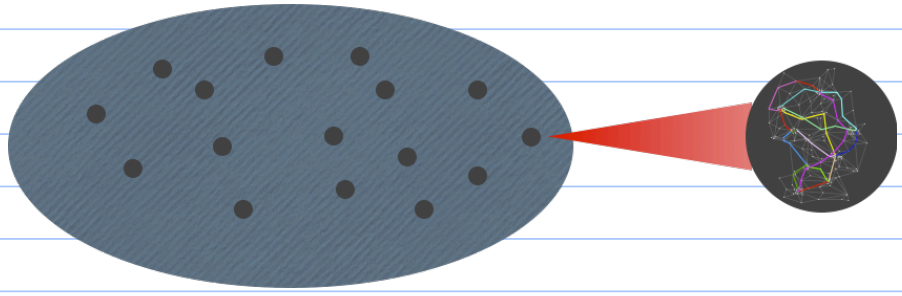
untere Schranken für den Optimalwert schränken die "Optimalitätslücke" ein

Also: Ist das Optimum unbekannt, so ergeben untere Schranken Qualitäts-Garantien für die Lösung von schweren Optimierungsproblemen

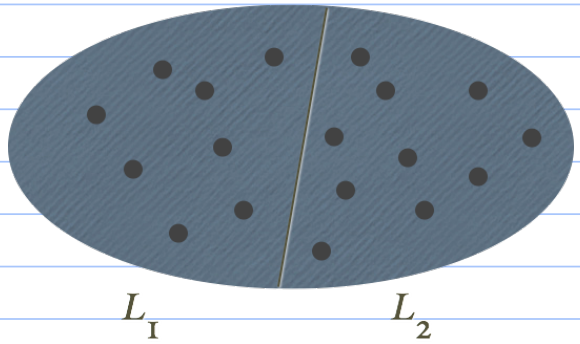
Branch & Bound

- illustriert am Disjunkte-Wege-Problem (vgl. Abschnitt 4.4)
- Stellen uns Lösungsraum (= Menge der zulässigen Lösungen) als Punktewolke vor

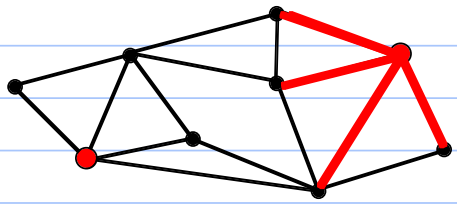
  Jeder Punkt repräsentiert eine zulässige Lösung



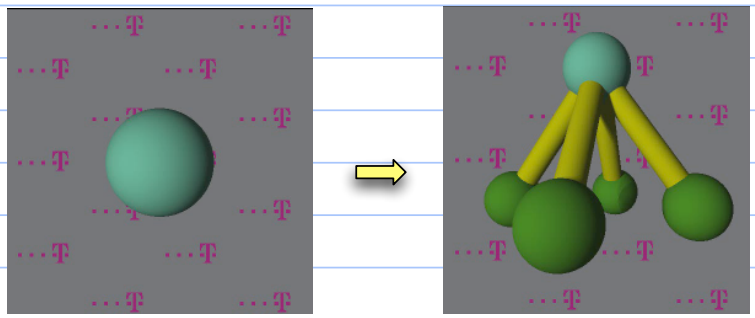
- Branching = Aufteilung der momentanen Lösungsmenge in ≥ 2 Teilmengen (nicht notwendig disjunkt)



- wird meistens als Baum dargestellt (Branch and Bound Tree)



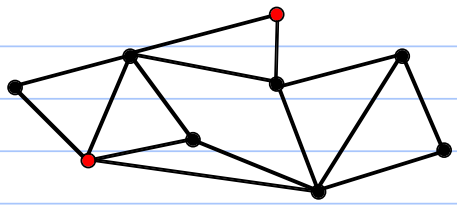
hier: teile Menge der Lösungen in 4 Teilmengen je nachdem welche rote Kante zur Verbindung der roten Terminale verwendet wird



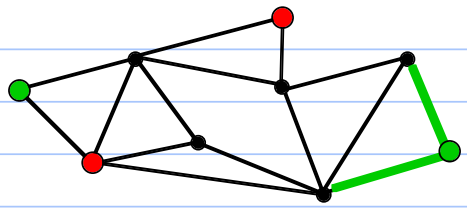
die Teilmengen sind dann im Baum Kinder der geteilten Menge

- dieser Prozess des Branching wird iteriert, dadurch entsteht der B&B Baum

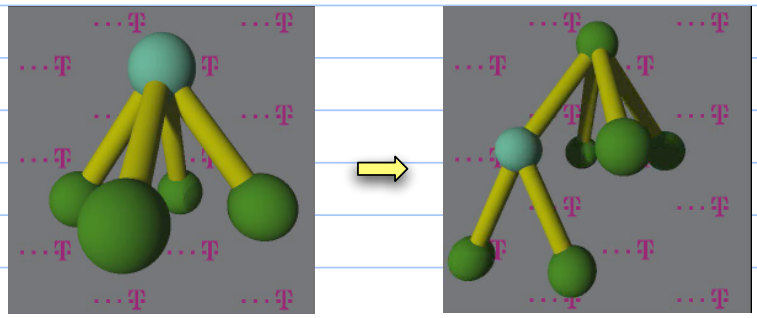
hier: Wahl der oberen roten Kante erzeugt dieses Teilproblem, in dem die obere rote Kante weggelassen wird (also für den verbindenden Weg reserviert wird) und das rote Terminal entsprechend verschoben wird



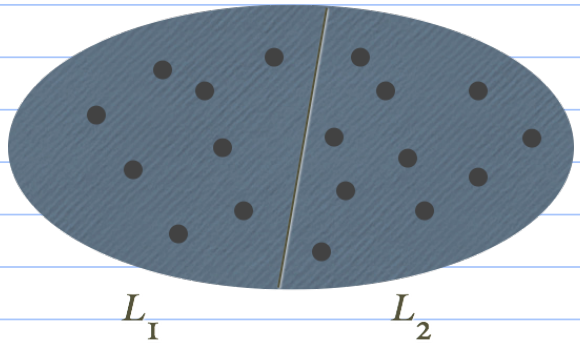
die Menge der zugehörigen zulässigen Lösungen kann dann bzgl einen eines anderen (oder desselben) Terminalpaares weiter zerlegt werden



hier: Wahl der grünen Terminale und Aufteilung in 2 Teilmenge in Abhängigkeit der gewählten grünen Kante



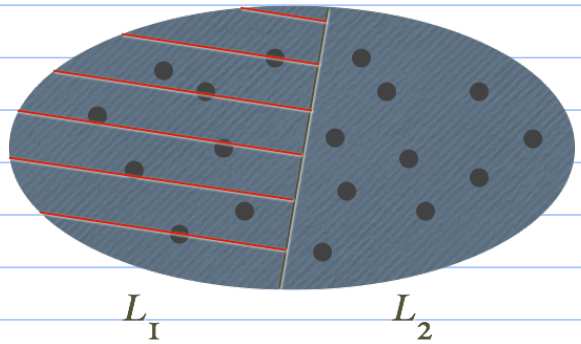
Bounding = Abschneiden von Ästen des B&B Baums durch Verwendung unterer Schranken



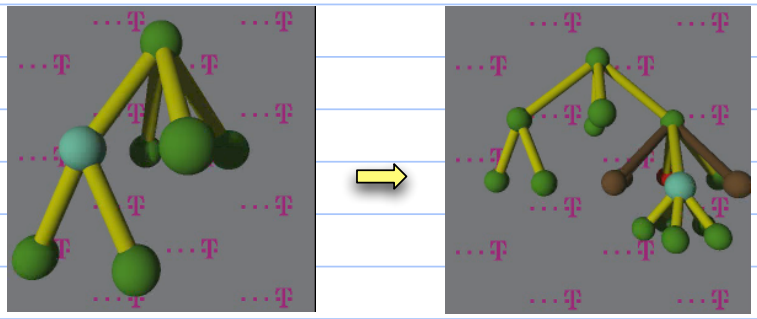
Annahme 1: wir kennen eine zulässige Lösung mit Kosten k

Annahme 2: wir kennen eine untere Schranke s für den Optimalwert in L_1

=> brauchen L_1 nicht zu untersuchen, wenn $s \geq k$



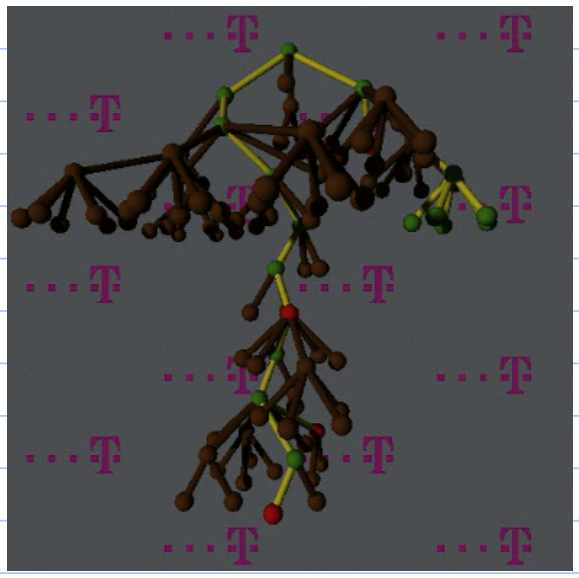
dieses Abschneiden von Ästen des B&B Baums bezeichnet man auch als Pruning, im Baum hier durch braune (verdorrte) Äste dargestellt. Der Knoten wird dann als fathomed bezeichnet.

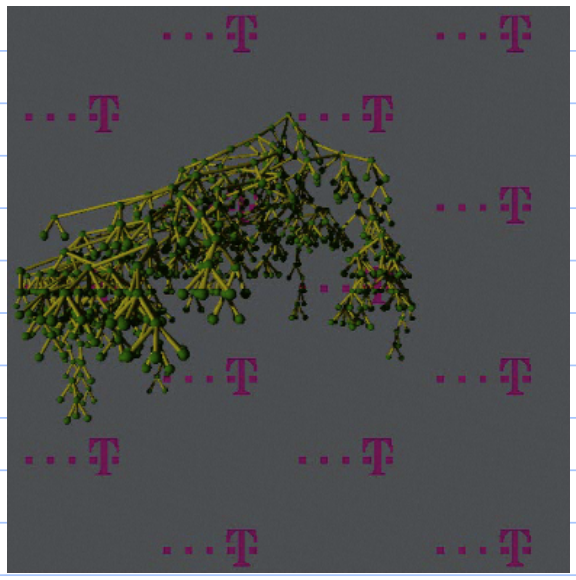


- Branching und Bounding wird verbunden mit
  - guten Auswahlstrategien zur Untersuchung der nächsten Knoten (= Teilmenge der zulässigen Lösungen) im B&B Baum
    - Tiefensuche
    - Breitensuche
    - Best-First-Search (gehe in Richtung bester (= kleinster) unterer Schranke)
    - Kombinationen davon
  - der Baum wird natürlich nur implizit verwaltet und nie explizit erzeugt

- Techniken zur Erzeugung guter unterer Schranken (nächstes Kapitel)
  - Lagrange Relaxation
  - LP-Relaxation (spzeziell bei IPs)
- Techniken zur Erzeugung zulässiger Lösungen (oberer Schranken) in Baumknoten

- Laufzeit ist exponentiell, hängt sehr von der Qualität der unteren Schranken ab





gute untere Schranken — kleiner B&B Baum

schlechte untere Schranken — riesiger B&B Baum

Schematische Darstellung von Branch and Bound

- Input
  - Instanz I eines Problems
- Output
  - zulässige Lösung $x \in S_I$ mit Gütegarantie in Form des Zielfunktionswertes $c(x)$ und einer unteren Schranke ℓ für den Optimalwert
- Ingredienzen
  - lower bounding strategy
  - branching strategy
  - search strategy
- Methode
  - 1. Arbeit in der Wurzel
    - betrachte eine abgewandelte, leichter zu lösende Instanz I' (Relaxation) zur Bestimmung einer unteren Schranke für I;

      berechne die Optimallösung x' von I' mit Zielfunktionswert z';

      if $x' \in S_I$ then return x' // x' ist optimal
    - setze $\ell := z'$ // anfängliche globale untere Schranke

- // initialisiere Datenstruktur D zur Verwaltung der noch zu untersuchenden Knoten des B&B Baums
- Füge I mit $\ell(I) := \ell$ in D ein
- Verwende Heuristiken zur Erzeugung zulässiger Lösungen
- setze $x^* :=$ beste gefundene Lösung
- setze u := bester gefundener Zielfunktionswert // anfängliche obere Schranke

2. Main loop

- **while** Güte $(u-\ell)/\ell$ nicht klein genug **and** noch Rechenzeit und Speicher verfügbar **do**
  - wähle nächsten zu untersuchenden Knoten v des B&B Baumes aus D // search strategy
  - **if** $\ell(v) \geq u$ **then** lösche v aus D // pruning
  - **else**
    - erzeuge die Kinder $v_1, \ldots, v_k$ von v // branching rule
    - // Vereinigung der Zulässigkeitsbereiche der Kinder = Zlässigkeitsbereich von v
    - **for** jedes Kind v_i **do**
      - berechne die Optimallösung x' (der Relaxation) des zugehörigen Teilproblems mit Zielfunktionswert z' // bounding rule
      - **if** $x' \in S_I$ and $z' < u$ **then**

        - $x^* := x'$ // Aktualisierung bester bekannter zulässiger Lösung
        - u := z' // Aktualisierung der oberen Schranke
      - else
        - if $z' < u$ then füge v_i mit $\ell(v_i) := z'$ in D ein // neues Teilproblem
    - lösche v aus D // v ist abgearbeitet
    - $\ell := \min \{ \ell(w) \mid w \text{ in } D \}$ // aktualisiere globale untere Schranke
- **return** x^* und ℓ

Branch and Bound bei IPs

- Für das Bounding liegt die Verwendung der LP Relaxation nahe
- Für das Branching liegt das Branching bzgl. fraktionaler Variablen in der LP Relaxation nahe

7.8 Beispiel (Das KNAPSACK Problem, vgl. ADM I)

KNAPSACK

Instanz

- n Gegenstände (items) mit Gewicht w_i und Wert (Profit) c_i

- ein Rucksack mit Kapazität W
- Aufgabe
  - Finde eine Teilmenge $S \subseteq \{ 1, \ldots, n \}$ mit
    - maximalem Wert $c(S) := \sum \{ c_j \mid j \in S \}$
    - Kapazität des Rucksacks wird nicht überschritten, d.h. $w(S) := \sum \{ w_j \mid j \in S \} \leq W$
- Eine IP Formulierung von KNAPSACK
  - Führe 0/1-Variable x_j ein mit $x_j = 1$, falls der Gegenstand j mitgenommen wird

$$\min \sum_j -c_j x_j$$
$$\sum_j w_j x_j \leq W$$
$$x_j \in \{ 0, 1 \}$$

7.9 Lemma (Optimallösungen der LP Relaxation von KNAPSACK)

- Man erhält eine Optimallösung der LP-Relaxation

$$\min \sum_j -c_j x_j$$
$$\sum_j w_j x_j \leq W$$
$$0 \leq x_j \leq 1$$

der IP Formulierung von KNAPSACK wie folgt

- sortiere und nummeriere die Gegenstände so, dass $c_1/w_1 \geq c_2/w_2 \geq \ldots c_n/w_n$ (größter Nutzen pro Gewichtseinheit zuerst)
- berechne in dieser Reihenfolge die kleinste Zahl k, so dass $w_1 + w_2 + \ldots + w_{k+1} > W$
- setze $x_1 = x_2 = \ldots = x_k = 1$

  $x_{k+1} = (W - w_1 - w_2 - \ldots - w_k)/w_{k+1}$

  $x_j = 0$ sonst

- Beweis durch Überprüfung der Bedingungen vom komplementären Schlupf
  - primal duales Paar ist gegeben durch

	c_1	$\ldots$	c_n	
u	w_1	$\ldots$	w_n	W
v_1	1			1
		$\ddots$		
v_n			1	1
	x_1	$\ldots$	x_n	

  - die Bedingungen vom komplementären Schlupf ergeben
    - (1) $x_j > 0 \Rightarrow w_j u + v_j = c_j$

- (2) $u > 0 \Rightarrow \sum_j w_j x_j = W$ (ist von der Lösung x erfüllt)
- (3) $v_j > 0 \Rightarrow x_j = 1$

Definiere nun zu x eine dual zulässige Lösung, die mit x die Bedingungen (1) und (3) erfüllt

- (3) $\Rightarrow v_{k+1} = v_{k+2} = \ldots = v_n := 0$

  $\Rightarrow$ (mit (1) für $j = k+1$) $w_{k+1}u = c_{k+1} \Rightarrow u = c_{k+1}/w_{k+1}$

  $\Rightarrow$ (mit (1) für $j = 1, \ldots k$) $w_j(c_{k+1}/w_{k+1}) + v_j = c_j$

  $\Rightarrow v_j := c_j - w_j(c_{k+1}/w_{k+1})$ für $j = 1, \ldots k$

  $\Rightarrow$ Werte für alle Dualvariablen definiert aus den Bedingungen (1) und (3)
- zeige noch: dies definiert eine dual zulässige Lösung

  dazu muss nur noch $v_j \geq 0$ gezeigt werden, d.h. $c_j - w_j(c_{k+1}/w_{k+1}) \geq 0$ für $j = 1, \ldots k$.

  Dies folgt aus $c_j/w_j \geq c_{k+1}/w_{k+1}$ für $j = 1, \ldots k$ ❑

Nutze allgemeines B&B Schema mit folgenden Ingredienzen

- lower bounding strategy = LP Relaxation gelöst mit Lemma 7.9
- branching strategy = branche auf fraktionaler Variablen x_{k+1}
- search strategy = best first

Daten der Instanz

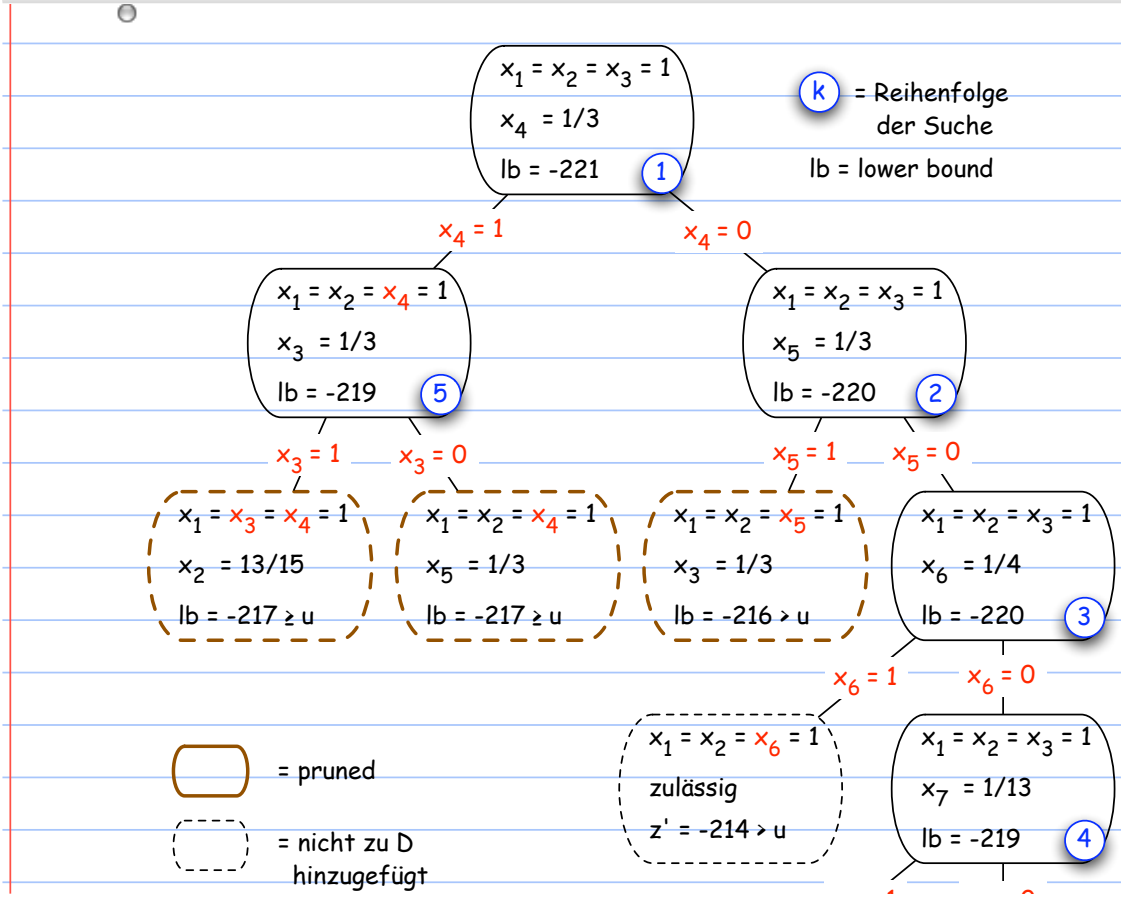
j	w_j	c_j	$\frac{c_j}{w_j}$
1	16	112	7
2	15	90	6
3	3	15	5
4	3	12	4
5	3	9	3
6	4	12	3
7	13	26	2

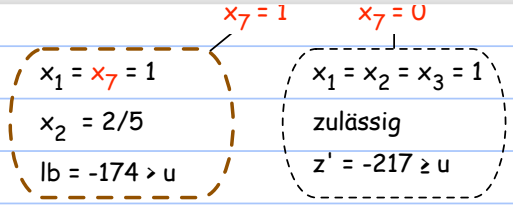
W = 35

heuristische Lösung $x_1 = x_2 = x_3 = 1$, $x_j = 0$ sonst $\Rightarrow$ obere Schranke $u = -217$

LP Relaxation ergibt $x_1 = x_2 = x_3 = 1, x_4 = 1/3, x_j = 0$ sonst $\Rightarrow$ untere Schranke $\ell = -221$

Branch and Bound Tree





Verwendung anderer Relaxationen als der LP-Relaxation

Dies ist möglich, z.B. durch Weglassen von Nebenbedingungen

=> Zulässigkeitsbereich wird größer => Minimum wird kleiner

7.10 Beispiel (TSP in Digraphen)

Eine IP Formulierung

Führe 0/1-Variable x_{ij} ein mit $x_{ij} = 1$ <=> Kante (i,j) ist in der Tour

$$\min \sum_{ij} c_{ij} x_{ij}$$

$$\sum_j x_{ij} = 1 \quad \text{für alle } i = 1, \ldots, n \qquad (7.1)$$

$$\sum_i x_{ij} = 1 \quad \text{für alle } j = 1, \ldots, n \qquad (7.2)$$

$$\sum_{i,j \in S} x_{ij} \leq |S|-1 \quad \text{für alle } \emptyset \neq S \subset \{1, \ldots, n\} \qquad (7.3)$$

$x_{ij} \in \{ 0, 1 \}$ (7.4)

- Die Cycle Cover Relaxation des TSP
  - Ergibt sich durch Weglassen der Bedingung (7.3).
  - Die verbleibenden Bedingungen beschreiben ein Zuordnungsproblem, wobei jedoch Kanten (i,i) nicht vorkommen dürfen. Dies kann in der Zielfunktion durch hohe Kosten c_{ii} berücksichtigt werden. Solche Zuordnungsprobleme sind effizient lösbar, z.B. mit der primal-dualen Methode, vgl. Abschnitt 6.5.

- Verwendung der Cycle Cover Relaxation in Branch and Bound Algorithmen
  - Nehme die Cycle Cover Relaxation als lower bounding strategy
  - Falls die optimale Zuordnung (x_{ij}) Bedingung (7.3) erfüllt, so ist sie eine Tour. Ansonsten branche wie folgt:
    - wähle den kürzesten Zykel bzgl. Kantenzahl und setze die Kanten darauf einzeln zu 0

      => pro Kante im Zykel ein Kind im B&B Baum

- **Hauptpunkte dieses Abschnittes**
  - Lagrange Relaxation ist eine wichtige Technik zur Erzeugung von "guten" unteren Schranken für IPs. Sie relaxiert Nebenbedingungen, bestraft aber ihre Verletzung in der Zielfunktion. Durch Variation der Strafkosten kann die untere Schranke verbessert werden.
  - Die systematische Verbesserung der Strafkosten führt zur Subgradientenoptimierung, einer Methode zur Maximierung einer nicht-differenzierbaren konkaven Funktion
  - Die dadurch erreichbare untere Schranke ist mindestens so gut wie bei der LP Relaxation, unter bestimmten Bedingungen jedoch gleich. Der Vorteil zur LP Relaxation liegt (Problem-abhängig) in der schnelleren (approximativen) Berechnung der Schranke durch kombinatorische Algorithmen.
  - Lagrange Relaxation ist eins der Arbeitspferde in Branch and Bound Algorithmen

- **Grundlagen der Lagrange Relaxation**
  - Gegeben sei das ganzzahlige Lineare Programm

    (P) min $c^T x$

    unter $Ax \geq b$ (k "schwere" Nebenbedingungen)

    $Bx \geq d$ (m-k "leichte" Nebenbedingungen)

x ganzzahlig

Relaxiere die "schweren" Nebenbedingungen $Ax \geq b$ und bestrafe ihre Verletzung in der Zielfunktion.

Führe dazu Lagrange-Multiplikatoren $\lambda_1, \ldots, \lambda_k$ für diese Nebenbedingungen ein. Sie bilden eine Art Dualvariable für diese Nebenbedingungen und erfüllen die Bedingungen

$a_i x \geq b_i \Rightarrow \lambda_i \geq 0$ (7.5)

$a_i x = b_i \Rightarrow \lambda_i$ nicht vorzeichenbeschränkt (7.6)

Für festes solches $\lambda = (\lambda_1, \ldots, \lambda_k)^T$ ist die Lagrange Relaxation (LR_λ) von (P) definiert als

(LR_λ) $\min c^T x + \lambda^T(b - Ax) =: L(\lambda, x)$

unter $Bx \geq d$

x ganzzahlig

$L(\lambda,x)$ wird Lagrange Funktion genannt, $\lambda = (\lambda_1, \ldots, \lambda_k)^T$ heißt auch Lagrange-Vektor und kann als Vektor von Strafkosten interpretiert werden.

Wir bezeichnen die Zulässigkeitsbereiche von (P) und (LR_λ) mit $S(P)$ und $S(LR_\lambda)$ und die zugehörigen Optimalwerte mit $z(P)$ und $z(LR_\lambda)$.

7.11 Lemma (Lagrange Relaxation liefert untere Schranken)

Für jeden Lagrange Vektor λ gilt

(1) $S(LR_\lambda) \supseteq S(P)$

(2) $z(LR_\lambda) \leq z(P)$

Beweis

(1) ist trivial, da Nebenbedingungen weggelassen werden

zu (2)

Sei x optimal bzgl. (P)

$\Rightarrow b_i - a_i x \leq 0$ bzw $b_i - a_i x = 0$ bei Gleichheitsrestiktionen

$\Rightarrow \lambda_i(b_i - a_i x) \leq 0$ für alle $i \Rightarrow \lambda^T(b - Ax) \leq 0$

$\Rightarrow z(P) = c^T x \geq c^T x + \lambda^T(b - Ax) \geq z(LR_\lambda)$ da $x \in S(P) \subseteq S(LR_\lambda)$ ❑

7.12 Lemma (Optimalitätsbedingungen)

Genügen x und λ den Bedingungen

(1) x ist optimal bzgl. (LR_λ)

(2) $a_i x \geq b_i$ bzw. $a_i x = b_i$ bei Gleichheitsrestriktionen

(3) $\lambda^T(b - Ax) = 0$

so ist x optimal bzgl. (P). Ist (3) nicht erfüllt, so ist x ε-optimal mit $\varepsilon = \lambda^T(b - Ax)$

Beweis

(1), (2) => $x \in S(P)$

=> $z(LR_\lambda) = c^Tx + \lambda^T(b - Ax) = c^Tx \geq z(P)$ wegen (3) und $x \in S(P)$

=> $z(LR_\lambda) = z(P)$ wegen Lemma 7.11.

Ist (3) nicht erfüllt, so ist $\lambda^T(b - Ax)$ der Fehlerterm ❑

Ziel der Lagrange Relaxation

Aufteilung der Restriktionen von (P) so, dass (LR_λ) im Verhältnis zu (P) leicht lösbar ist

$z(P) - z(LR_\lambda)$ möglichst klein machen (Dualitätslücke der Lagrange Relaxation)

d.h. $L(\lambda) := z(LR_\lambda)$ möglichst groß machen durch Variation der Lagrange Multiplikatoren

=> dies führt zu Optimierungsproblem $\max_\lambda L(\lambda)$

Dieses Optimierungsproblem muss bei Verwendung in B&B nicht optimal gelöst werden, es reicht ein guter Wert von $L(\lambda)$, da jeder solche Wert eine untere Schranke für $z(P)$ liefert.

Lagrange Relaxation des symmetrischen TSP mittels 1-Bäumen

IP Formulierung des symmetrischen TSP

Führe 0/1-Variable x_e ein mit $x_e = 1$ <=> Kante e ist in der Tour

(P) min $\sum_e c_e x_e$

$x(\delta(i)) = 2$ für alle $i = 1, \ldots, n$ (7.7)

$x(S) \leq |S|-1$ für alle $\emptyset \neq S \subseteq \{ 2, \ldots, n \}$ (7.8)

beachte: $S \subseteq \{ 2, \ldots, n \}$ reicht, um Kurzzykel auszuschließen

$x_e \in \{ 0, 1 \}$ (7.9)

Dabei ist $x(S) := \sum_{e = ij,\, i,j \in S} x_e$ und $x(\delta(i)) := \sum_{e \in \delta(i)} x_e$

Variation von (P) führt zu (LR_λ)

spalte (7.7) auf in

$\sum_e x_e = n$ (7.10) redundant in (P)

$x(\delta(i)) = 2$ für $i = 2, \ldots, n$ (7.11)

$x(\delta(1)) = 2$ (7.12)

Definiere (LR_λ) durch die Relaxation von (7.11)

(LR_λ) min $\sum_e c_e x_e + \sum_{i=2,\dots,n} \lambda_i(2 - x(\delta(i))$

unter (7.8), (7.9), (7.10), (7.12)

Beachte: (7.10) ist nicht redundant in (LR_λ)

Kombinatorische Struktur der zulässigen Lösungen von (LR_λ)

7.13 Lemma (zulässige Lösungen von (LR_λ) sind 1-Bäume)

x ist zulässige Lösung von (LR_λ) <=> x ist ein 1-Baum, d.h.

x ist ein spannender Baum auf der Knotenmenge { 2, ..., n }

mit 2 zusätzlichen Kanten vom Knoten 1 aus

Beweis

"=>"

x zulässige Lösung von (LR_λ)

(7.9), (7.10), (7.12) => x hat n-2 Kanten auf den Knoten 2, ..., n

(7.8) => x ist zusammenhängend

ADM I => ein zusammenhängender Graph mit n-2 Kanten auf n-1 Knoten ist ein spannender Baum

(7.12) => zusätzlich 2 Kanten vom Knoten 1 aus

=> x ist 1-Baum

"<="

jeder 1-Baum erfüllt die Bedingungen (7.8), (7.9), (7.10), (7.12) ❑

Die Lagrangefunktion $L(\lambda,x)$

$L(\lambda,x) = \sum_e c_e x_e + \sum_{i=2,\dots,n} \lambda_i(2 - x(\delta(i))$, λ_i nicht vorzeichenbeschränkt

=> o.B.d.A. λ_i durch $-\lambda_i$ ersetzen (bessere kombinatorische Interpretation)

=> $L(\lambda,x) = \sum_e c_e x_e + \sum_{i=2,\dots,n} \lambda_i(x(\delta(i) - 2)$

mit $x(\delta(i)) - 2$ = Abweichung vom angestrebten Grad 2 des Knoten i

Mit $\lambda_1 := 0$ ergibt sich

$L(\lambda,x) = \sum_e c_e x_e + \sum_{i=1,\dots,n} \lambda_i(x(\delta(i) - 2)$

$= \sum_e c_e x_e + \sum_{i=1,\dots,n} \lambda_i x(\delta(i)) - 2\sum_{i=1,\dots,n} \lambda_i$

$= \sum_e c_e x_e + \sum_{e=ij} (\lambda_i + \lambda_j) x_e - 2\sum_{i=1,\dots,n} \lambda_i$

$= \sum_{e=ij} (c_e + \lambda_i + \lambda_j) x_e - 2\sum_{i=1,\dots,n} \lambda_i$

Dies entspricht neuen Kantenkosten $c_e' = c_e + \lambda_i + \lambda_j$ für $e = ij$ abzüglich eines konstanten Terms $2\sum_{i=1,\dots,n}$

λ_i

Interpretation der Lagrange Relaxation

Relaxiertes Problem

= Ermittlung eines 1-Baums mit minimalem Gewicht bzgl. der Kantenkosten $c_e + \lambda_i + \lambda_j$ für e = ij

Variation der Lagrange Multiplikatoren λ_i

= Variation der Kantenkosten c_e über Knotenbewertungen λ_i

Diese Variation der Kantenkosten hat keinen Einfluss auf die Optimalität einer Tour, verändern aber den 1-Baum

denn:

$$\sum_{e=ij} (c_e + \lambda_i + \lambda_j) x_e - 2 \sum_{i=1,\dots,n} \lambda_i = \sum_e c_e x_e \quad \text{wenn } x \text{ eine Tour ist}$$

Ist der minimale 1-Baum eine Tour, so ist diese Tour optimal für (P) nach Lemma 7.12, da $\lambda^T(b - Ax) = 0$ bei einer Tour

Einen minimalen 1-Baum findet man in polynomialer Zeit durch

(1) Ermittlung eines MST auf den Knoten 2, ..., n mit den Algorithmen aus ADM I (Kruskal oder Prim)

(2) Wahl der beiden billigsten Kanten vom Knoten 1 aus

Algorithmus zur Verbesserung der unteren Schranke (Variation der λ_i)

Input

Graph G = (V, E) mit V = { 1, ..., n }

Kantenkosten c_e

Output

optimale Tour oder 1-Baum mit "guter" unterer Schranke $z(LR_\lambda)$

Methode

// Initialisierung der λ_i

setze $\lambda_i := 0$ für jeden Knoten i

// Initialisierung einer Schrittweite $w > 0$ für die Variation der λ_i

setze w := 1

repeat

ermittle minimalen 1-Baum x bzgl der Kantenkosten $c_{ij} + \lambda_i + \lambda_j$

if x ist Tour then return x // x ist optimale Tour

// Variation der λ_i

for alle Knoten $i \neq 1$ do

bestimme den Grad d_i von Knoten i

if $d_i \neq 2$ then $\lambda_i := \lambda_i + (d_i - 2)w$

variiere ggf. die Schrittweite w

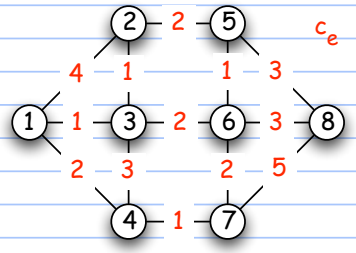
until $z(LR_\lambda) = z(x)$ ist "gut" genug

return bestes bisher gefundenes x und das zugehörige λ

7.14 Beispiel (1-Baum Relaxation des symmetrischen TSP)

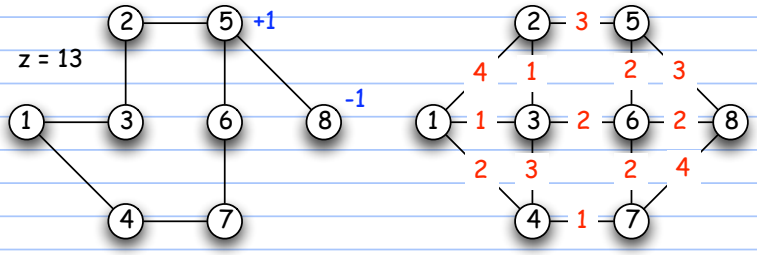
Schrittweite w immer als 1 gewählt

Graph mit Kantenkosten



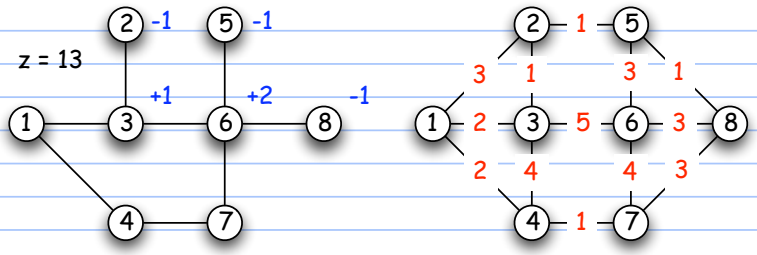
Iteration 1

minimaler 1 Baum, Variation der λ_i und neue Kantenkosten



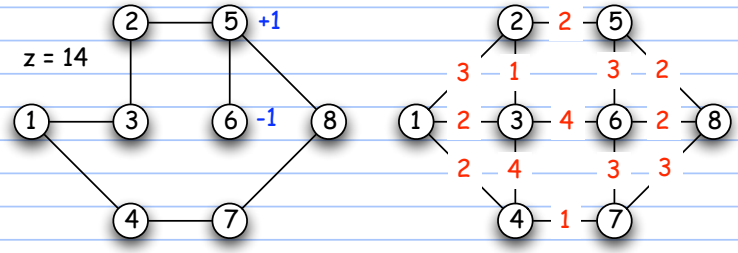
Iteration 2

minimaler 1 Baum, Variation der λ_i und neue Kantenkosten



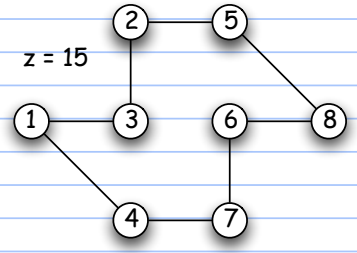
Iteration 3

minimaler 1 Baum, Variation der λ_i und neue Kantenkosten



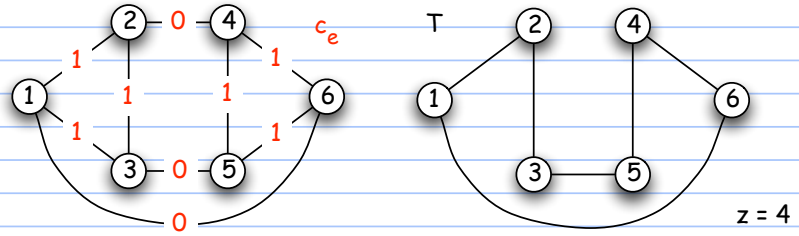
Iteration 4

minimaler 1 Baum ist Tour => optimale Tour konstruiert



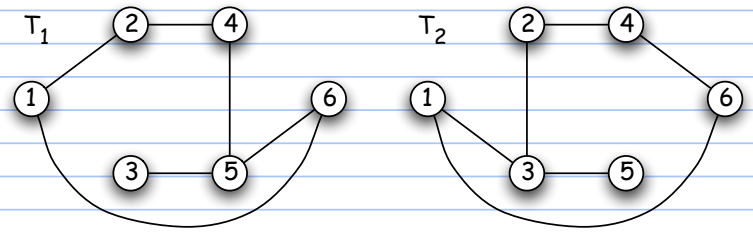
7.15 Beispiel (i.A. gibt es kein λ so dass der optimale 1-Baum eine Tour ist)

Graph mit Kantenkosten und optimale Tour



Claim: Für jede Wahl von λ_i (mit $\lambda_1 = 0$) ist keine Tour bzgl. $c_{ij} + \lambda_i + \lambda_j$ ein minimaler 1-Baum

Betrachte die 1-Bäume



Die Werte der beiden 1-Bäume bzgl. $c_{ij} + \lambda_i + \lambda_j$ ergeben sich als

Wert von $T_1 = 3 + 2\lambda_2 + 1\lambda_3 + 2\lambda_4 + 3\lambda_5 + 2\lambda_6 =: z_1$

Wert von $T_2 = 3 + 2\lambda_2 + 3\lambda_3 + 2\lambda_4 + 1\lambda_5 + 2\lambda_6 =: z_2$

Der Wert einer optimalen Tour bzgl. $c_{ij} + \lambda_i + \lambda_j$ ergibt sich als

$4 + 2\lambda_2 + 2\lambda_3 + 2\lambda_4 + 2\lambda_5 + 2\lambda_6 =: z_0$

$\Rightarrow z_0 - z_1 = 1 + \lambda_3 - \lambda_5$ und $z_0 - z_2 = 1 - \lambda_3 + \lambda_5$

$\Rightarrow$ entweder $z_0 > z_1$ oder $z_0 > z_2$

denn aus $z_0 < z_1$ und $z_0 < z_2$ folgt $1 + \lambda_3 - \lambda_5 < 0$ und $1 - \lambda_3 + \lambda_5 < 0$

$\Rightarrow \lambda_3 - \lambda_5 > 1$ und $-\lambda_3 + \lambda_5 > 1$, Widerspruch ❑

Beachte: Der hier beim TSP beobachtete Fall $\max_\lambda L(\lambda) \neq z(P)$ ist der Regelfall. Die Lagrange-Relaxation liefert i.A. nur untere Schranken für $z(P)$, die allerdings oft gut in einen Branch & Bound Algorithmus einbettbar sind.

Zu weiteren Informationen über Lagrange Relaxationen des TSP siehe

E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, eds.
The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization
John Wiley & Sons, New York, 1985.

Bestimmung von $\max_\lambda L(\lambda)$ mit Subgradientenverfahren

$\max_\lambda L(\lambda) = \max_\lambda \min_x L(\lambda,x) = \max_\lambda \min \{ L(\lambda,x) \mid x \in S(LR_\lambda) \}$

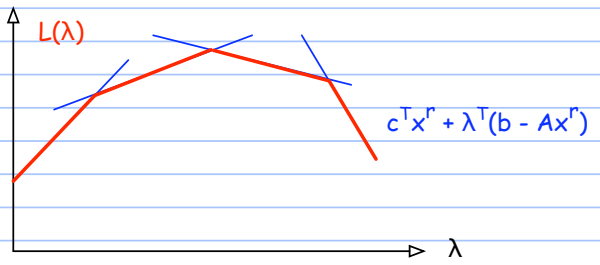
Das Subgradientenverfahren nutzt die Tatsache, dass $S(LR_\lambda)$ wegen der Ganzzahligkeit von x in der Regel endlich ist. Dies nehmen wir im Folgenden an.

$S(LR_\lambda)$ endlich $\Rightarrow$ können $S(LR_\lambda)$ schreiben als $S(LR_\lambda) = \{ x^1, x^2, \ldots, x^R \}$

$\Rightarrow L(\lambda) = \min \{ c^T x^r + \lambda^T (b - Ax^r) \mid r = 1, \ldots, R \}$

$\Rightarrow L(\lambda)$ ist Minimum von endlich vielen affin linearen Funktionen $c^T x^r + \lambda^T (b - Ax^r)$ in λ

$\Rightarrow L(\lambda)$ ist stückweise linear und konkav, aber i.A. nicht differenzierbar



Subgradientenverfahren

~ Gradientenverfahren der Maximierung einer konkaven stetig differenzierbaren Funktion $f : \mathbb{R}^n \to \mathbb{R}^1$

Gradient und Subgradient

Gradient einer stetig differenzierbaren Funktion in u

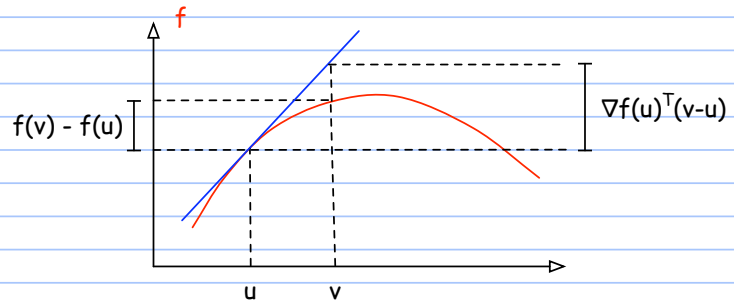
= Vektor der partiellen Ableitungen in u:

$$\nabla f(u) = \left(\frac{\partial f}{\partial x_1}(u), \ldots, \frac{\partial f}{\partial x_n}(u) \right)$$

Dann ist aus der Analysis bekannt:

f ist konkav <=>

für alle v, u gilt $f(v) - f(u) \leq \nabla f(u)^T (v-u)$



Subgradient einer stetigen konkaven Funktion in u

= Vektor d mit $f(v) - f(u) \leq d^T(v-u)$ für alle v

Die Menge der Subgradienten in u heißt das Subdifferential von f in u und wird mit $\partial f(u)$ bezeichnet

Dann gilt: f differenzierbar in u => $\partial f(u) = \{ \nabla f(u) \}$

Bedingungen für Maximalpunkte einer konkaven Funktion

Der stetig differenzierbare Fall

Aus der Analysis ist bekannt:

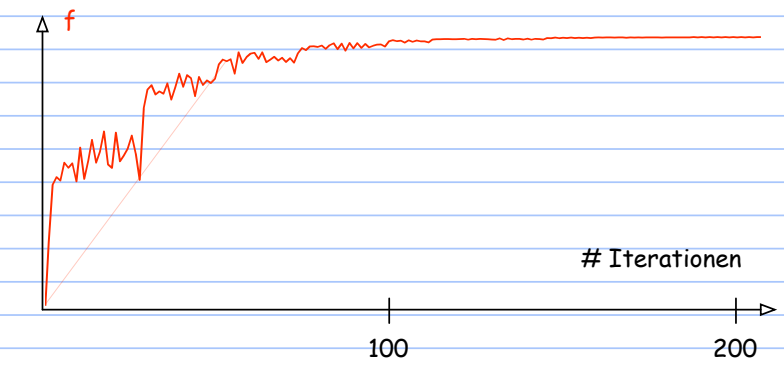
λ^* ist Maximalpunkt von f <=> $\nabla f(\lambda^*) = 0$

- Der nicht differenzierbare Fall
  - 7.16 Lemma (Bedingung für Maximalpunkt einer stetigen konkaven Funktion)
    - Für eine stetige konkave Funktion $f : \mathbb{R}^n \to \mathbb{R}^1$ gilt
    - λ^* ist Maximalpunkt von f <=> $0 \in \partial f(\lambda^*)$
    - Beweis
      - "<="
        - sei $0 \in \partial f(\lambda^*)$
        - => $0 = 0^T(v - \lambda^*) \geq f(v) - f(\lambda^*)$ für alle v => λ^* ist Maximalpunkt von f
      - "=>"
        - sei λ^* ist Maximalpunkt von f
        - => $0 = 0^T(v - \lambda^*) \geq f(v) - f(\lambda^*)$ für alle v => $0 \in \partial f(\lambda^*)$ ❑

- Generisches Subgradientenverfahren
  - Input
    - stetige konkave Funktion $f : \mathbb{R}^n \to \mathbb{R}^1$

  - Output
    - Maximalpunkt λ^* oder Punkt λ mit "gutem" Wert $f(\lambda)$
  - Methode
    - wähle Startwert u_0
    - initialisiere Zähler $i := 0$
    - repeat
      - if $0 \in \partial f(u_i)$ then return u_i // u_i ist Maximalpunkt
      - // hierauf kann auch verzichtet werden falls der Test "$0 \in \partial f(u_i)$" aufwändig ist
      - bestimme Subgradient $d_i \in \partial f(u_i)$ und Schrittweite $w_i > 0$
      - setze $u_{i+1} := u_i + w_i \cdot d_i$
      - $i := i+1$
    - until Rechenzeit zuende oder kaum noch Fortschritt
    - return besten Punkt der Folge $u_0, \ldots, u_i$

- Beispiel eines typischen Laufes



Lauf zeigt, dass i.A. keine Monotonie gegeben ist

Hauptingredienzen des Subgradientenverfahren

bestimme Schrittweite $w_i > 0$

theoretisch gelöst durch Satz von Polyak, aber praktisch dennoch schwierig, erfordert i.A. viele Experimente

7.17 Satz (Satz von Polyak 1967)

Sei $f : \mathbb{R}^n \to \mathbb{R}^1$ konkav und stetig und f nehme ihr Maximum im Punkt λ^* an.

Sei $(w_i)_{i \in N}$ eine Folge von Schrittweiten mit

(1) $w_i \geq 0$ für alle i

(2) $(w_i)_{i \in N}$ ist eine monoton fallende Nullfolge

(3) die Reihe $\sum w_i$ ist divergent

Dann gilt für die im Subgradientenverfahren erzeugte Folge von Punkten u_i

$\lim_{i \to \infty} f(u_i) = f(\lambda^*)$

ohne Beweis ❑

Dieser Satz sichert Konvergenz unter relativ schwachen Bedingungen, die bei konkreten Berechnungen leicht einzuhalten sind. Das Problem ist die Steuerung der Konvergenzgeschwindigkeit. Bei der Verwendung in B&B ist dies aber auch nicht so wesentlich.

bestimme Subgradient $d_i \in \partial f(u_i)$

dies ist einfacher, Subgradienten bekommt man bei der Lagrange Relaxation geschenkt

7.18 Lemma (Subgradienten bei der Lagrange Relaxation)

Sei x^* optimale Lösung von (LR_λ) in $\lambda = u$.

Dann ist $b - Ax^*$ Subgradient von $L(\lambda) = \min_x L(\lambda,x)$ in $\lambda = u$, d.h. $b - Ax^* \in \partial f(u)$.

Beweis durch Überprüfung der Definition

$L(v) - L(u) = \min_x L(v,x) - \min_x L(u,x)$

$= \min_x L(v,x) - L(u,x^*)$ da x^* optimal für (LR_u) ist

$\leq L(v,x^*) - L(u,x^*)$ da x^* zulässig für (LR_v) ist

$= (c^Tx^* + v^T(b - Ax^*)) - (c^Tx^* + u^T(b - Ax^*))$

$= (v^T - u^T)(b - Ax^*) = (b - Ax^*)^T(v - u)$ ❑

Bemerkung: Bei der 1-Baum Relaxation des symmetrischen TSP ergibt sich (nach Übergang von λ_i zu $-\lambda_i$) $x(\delta(i)) - 2$ als Subgradient. Die Veränderung der Multiplikatoren λ_i erweist sich damit als Spezialfall des Subgradientenverfahren.

Lagrange Relaxation vs. LP Relaxation

Es besteht eine Beziehung zwischen dem Optimalwert der Lagrange Relaxation und dem Wert der LP Relaxation

eines IP.

Wir betrachten dazu:

Das Ausgangsproblem

(P) $\min c^Tx$

unter $Ax \geq b$

$Bx \geq d$

x ganzzahlig

keine Vorzeichenbedingungen an x, diese sind ggf. in die Nebenbedingungen integriert

Die Lagrange Relaxation von (P)

(LR_λ) $\min c^Tx + \lambda^T(b - Ax) = \min_x L(\lambda, x) = L(\lambda)$

unter $Bx \geq d$

x ganzzahlig

Die LP Relaxation von (P)

(LP) $\min c^Tx$

unter $Ax \geq b$

$Bx \geq d$

x beliebig

mit Optimalwert $z(LP)$

7.19 Satz (Beziehung zwischen Lagrange Relaxation und LP Relaxation)

$\max_\lambda L(\lambda) \geq z(LP)$

Gleichheit gilt falls das von $Bx \geq d$ erzeugte Polyeder ganzzahlig ist (also die Ganzzahligkeitsbedingung in (LR_λ) weggelassen werden kann).

Beweis

wird hier geführt für Nebenbedingungen der Form $Ax \geq b$ ($\Rightarrow \lambda \geq 0$), er folgt für Gleichheitsrestriktionen (λ beliebig) entsprechend.

$$\max_{\lambda \geq 0} L(\lambda) = \max_{\lambda \geq 0} \min_{\substack{x \\ Bx \geq d \\ x\ gzz}} L(\lambda,x) \quad \uparrow \quad = \max_{\lambda \geq 0} \min_{\substack{x \\ Bx \geq d}} L(\lambda,x)$$

Bx ≥ d erzeugt ganzzahliges Polyeder, sonst gilt hier ≥

$$= \max_{\lambda \geq 0} \min_{\substack{x \\ Bx \geq d}} (c^T x + \lambda^T (b - Ax))$$

$$= \max_{\lambda \geq 0} [\lambda^T b + \min_{\substack{x \\ Bx \geq d}} (c^T - \lambda^T A)x] = \max_{\lambda \geq 0} [\lambda^T b + \max_{\substack{y \geq 0 \\ B^T y = c - A^T \lambda}} d^T y]$$

LP Dualität

$$= \max_{\substack{\lambda \geq 0 \\ y \geq 0 \\ B^T y = c - A^T \lambda}} [b^T \lambda + d^T y] = \min_{\substack{x \text{ beliebig} \\ Ax \geq b \\ Bx \geq d}} c^T x$$

LP Dualität

$= z(LP)$ ❑

7.20 Bemerkung

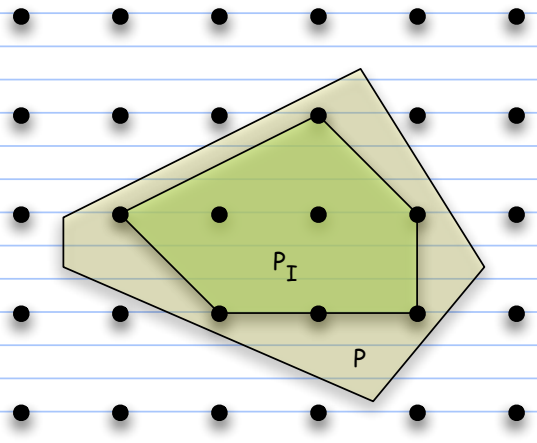
- Die 1-Baum Relaxation entspricht nach Satz 7.19 der LP-Relaxation des TSP-Polytopes.
- Da LPs im Prinzip in polynomialer Zeit lösbar sind (innere Punkte Methoden) scheint die LP-Relaxation vorzuziehen sein, falls $Bx \geq d$ ein ganzzahliges Polyeder definiert. Dennoch ist in der Praxis sehr oft das Subgradientenverfahren vorzuziehen, da es wesentlich schneller ist (oft kann $L(\lambda)$ kombinatorisch berechnet werden) und meist Näherungswerte für $\max_\lambda L(\lambda)$ reichen.

Hauptpunkte dieses Abschnittes

- Schnittebenenverfahren einführen als ein weiteres Verfahren um IPs exakt zu lösen.
- Nachweis, dass dies ein im Prinzip ein endliches Verfahren ist.
- Bei diesem Nachweis lernen wir einiges über ganzzahlige Polytope (Gomory-Chvátal-Schnitte, Chvátal Hülle)

Die ganzzahlige Hülle eines Polyeders

- Die ganzzahlige Hülle (integer hull) P_I eines Polyeders P ist die konvexe Hülle aller ganzzahligen Punkte in P.



Ein Polyeder P heißt ganzzahlig, wenn alle Ecken ganzzahlig sind.

Dann gilt für Polytope P

(1) P ist ganzzahlig <=> $P = P_I$

(2) Ganzzahlige Optimierung über P <=> lineare Optimierung über P_I

Daher ist man an linearen Beschreibungen (= Beschreibung als Ungleichungssystem) von P_I interessiert

Leider ist P_I im Allgemeinen kein Polyeder mehr!

Ein Beispiel ensteht für

$$P := \{(y, x) \in \mathbb{R}^2 \mid \frac{y}{x} \leq \sqrt{2}\}$$

(Übung)

Man kann jedoch zeigen, dass P_I für rationale Polyeder P wieder ein Polyeder ist. Wir zeigen dies hier für rationale Polytope P.

Ein Polyeder $P = \{ x \in \mathbb{R}^n \mid Ax \leq b \}$ heißt rational, wenn alle Einträge von A und b rationale Zahlen sind. Im Weiteren werden alle Polyeder als rational vorausgesetzt, in den Sätzen wird es der Vollständigkeit halber immer als Voraussetzung aufgeführt.

Kriterien für die Existenz zulässiger Punkte und gültige Ungleichungen

Diese Kriterien sind alternative Formulierungen des Farkas Lemma (Lemma 4.5).

Eine Ungleichung $w^T x \leq t$ heißt gültig für das Polyeder P, wenn alle Punkte $x \in P$ die Ungleichung erfüllen.

7.21 Lemma (Farkas Lemma für die Existenz zulässiger Lösungen)

Für ein Polyeder $P = \{ x \in \mathbb{R}^n \mid Ax \leq b \}$ gilt:

(1) $P \neq \emptyset$ <=> $y^T b \geq 0$ für alle $y \in \mathbb{R}^m$ mit $y \geq 0$ und $y^T A = 0$

(2) $P = \emptyset$ <=> es gibt $y \in \mathbb{R}^m$, $y \geq 0$ mit $y^T A = 0$ und $y^T b \leq -1$

(3) $P = \emptyset$ <=> die Ungleichung $0^T x \leq -1$ ergibt sich als nicht-negative Linearkombination der Ungleichungen in $Ax \leq b$

Beweis

zu (1)

"=>"

Betrachte das LP $\max \{ 0^T x \mid Ax \leq b \}$

$P \neq \emptyset$ => jedes $x \in P$ ist Optimallösung des LP

Dualitätssatz => das duale LP hat eine optimale Lösung und

$0 = \max\{ 0^T x \mid Ax \le b \} = \min\{ y^T b \mid y^T A = 0, y \ge 0 \}$

$\Rightarrow y^T b \ge 0$ für alle $y \ge 0$ mit $y^T A = 0$

"<="

Betrachte das LP $\min\{ y^T b \mid y^T A = 0, y \ge 0 \}$

0 ist zulässige Lösung dieses LP

Vorraussetzung => die Zielfunktion $y^T b$ ist nach unten durch 0 beschränkt

Dualitätssatz => P hat eine optimale Lösung, also insbesondere eine zulässige Lösung

zu (2)

durch Negation von (1) folgt

$P = \emptyset \Leftrightarrow$ es gibt $y' \in \mathbb{R}^m, y' \ge 0$ mit $(y')^T A = 0$ und $(y')^T b < 0$.

Sei $g := (y')^T b < 0$

Mit $y := y'/|g|$ folgt

$P = \emptyset \Leftrightarrow$ es gibt $y \in \mathbb{R}^m, y \ge 0$ mit $y^T A = 0$ und $y^T b \le -1$

zu (3)

"<="

klar

"=>"

nehme das y aus (2) und multipliziere $Ax \le b$ damit von links =>

$0^T x = y^T A x \le y^T b \le -1 \Rightarrow 0^T x \le -1$ ❑

7.22 Lemma (Farkas Lemma für gültige Ungleichungen)

Für ein nichtleeres Polyeder $P = \{ x \in \mathbb{R}^n \mid Ax \le b \}$ sind folgende Aussagen äquivalent:

(1) $w^T x \le t$ ist eine gültige Ungleichung für P

(2) Es gibt $y \in \mathbb{R}^m, y \ge 0$ mit $y^T A = w^T$ und $y^T b \le t$

Beweis

(1) => (2)

Betrachte das LP $\max\{ w^T x \mid Ax \le b \}$

$P \ne \emptyset$, $w^T x \le t$ => das LP hat eine optimale Lösung

Dualitätssatz => das duale LP hat eine optimale Lösung y^* und

$t \ge \max\{ w^T x \mid Ax \le b \} = \min\{ y^T b \mid y^T A = w^T, y \ge 0 \} = (y^*)^T b$

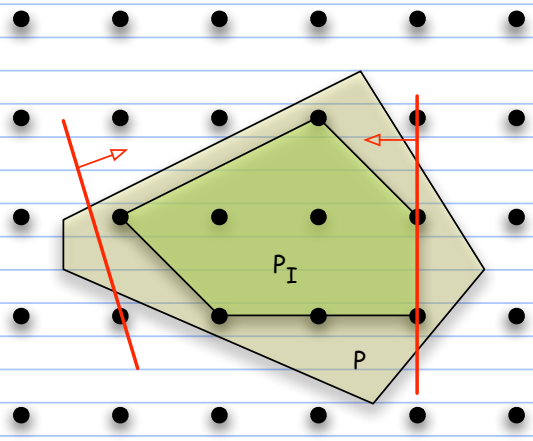
=> y^* erfüllt (2)

(2) => (1)

$Ax \le b,\ y \ge 0 \Rightarrow\ w^T x = (y^T A)x = y^T(Ax) \le y^T b \le t$ ❑

Schnittebenen (cutting planes) und Schnittebenenverfahren

Idee: Schneide durch Hyperebenen Teile eines Polyeders P ab, aber ohne Punkte aus P_I abzuschneiden, d.h. sie sind gültig für P_I (und vielleicht sogar Stützhyperebenen oder Facetten). Eine solche Hyperebene heißt Schnittebene (cutting plane). Eine Schnittebene H, die einen Punkt $x^* \in P - P_I$ abschneidet, heißt x^* separierende Hyperebene.



Schnittebenenverfahren (Idee)

Input

Ganzahliges lineares Programm (IP)

$\min \{ c^T x,\ x \in P_I \}$ mit P = Polyeder der LP Relaxation von (IP)

Output

Optimallösung von (IP)

Methode

repeat forever

löse das LP $\min \{ c^T x,\ x \in P \}$

sei x^* die Optimallösung des LP

if x^* ist ganzzahlig then return x^* // x^* ist Optimallösung von (IP)

berechne Schnittebene, die x^* von P abschneidet und gültig für P_I ist // separating hyperplane

sei H der zugehörige Halbraum, der P_I enthält

setze $P := P \cap H$

- Offensichtliche Fragen
  - (1) wie beweist man, dass eine Ungleichung eine Schnittebene ist?
    (2) terminiert das Schnittebenenverfahren überhaupt?
    (3) wie berechnet man zu gegebenem $x^* \in P - P_I$ eine x^* separierende Hyperebene?
  - Hier werden wir vor allem (1) und (2) behandeln und insofern positiv beantworten, als dass man Beweise für (1) angeben kann und es immer eine endliche Menge von Schnittebenen einer speziellen Gestalt gibt, so dass am Ende $P = P_I$ gilt.
  - (3) hängt sehr vom jeweiligen Problem ab, dazu mehr in Kapitel 8.

- **Schnittebenenbeweise**
  - Für Polytope $P = \{ x \in \mathbb{R}^n \mid Ax \leq b \}$ kann ein Beweis für die Gültigkeit einer Ungleichung $w^T x \leq t$ durch das Farkas Lemma (Lemma 7.21) gegeben werden. Für Schnittebenen ist es komplizierter.

- 7.23 Beispiel (Beispiel eines Schnittebenenbeweises)
  - Betrachte das Ungleichungssystem

$$2x_1 + 3x_2 \leq 27 \quad (1)$$

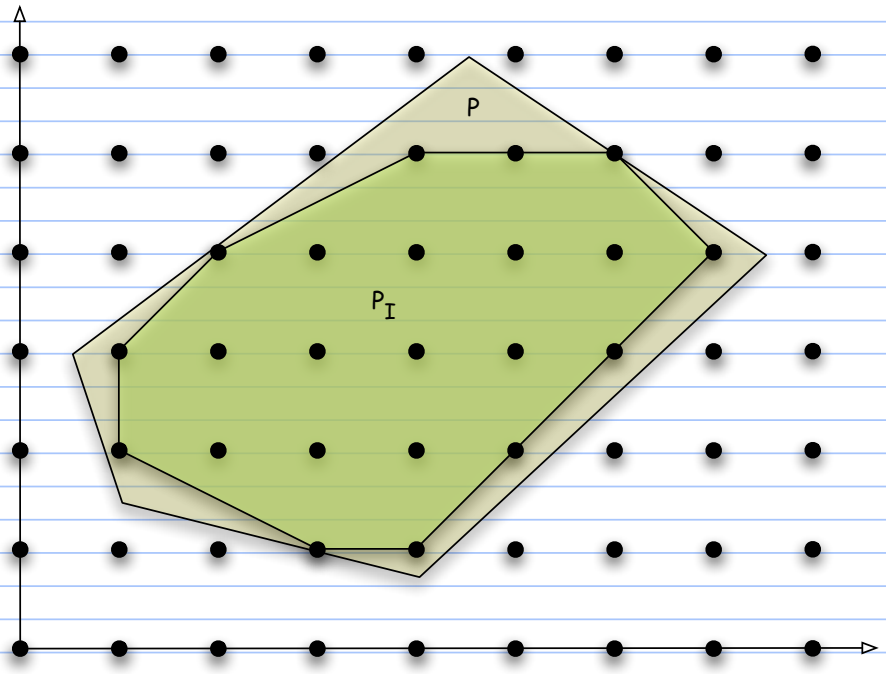
$$2x_1 - 2x_2 \leq 7 \quad (2)$$

$$-6x_1 - 2x_2 \leq -9 \quad (3)$$

$$-2x_1 - 6x_2 \leq -11 \quad (4)$$

$$-6x_1 + 8x_2 \leq 21 \quad (5)$$

- Das zugehörige Polytop P und seine ganzzahlige Hülle



$x_2 \leq 5$ ist eine gültige Ungleichung für P_I

Wie kann man das aus den Ungleichungen für P ableiten?

Multipliziere (5) mit 1/2

=> $-3x_1 + 4x_2 \leq 21/2$

=> $-3x_1 + 4x_2 \leq \lfloor 21/2 \rfloor = 10$ ist gültig für P_I (da links nur ganzzahlige Koeffizienten)

=> neue Ungleichung $-3x_1 + 4x_2 \leq 10$ (6) für P_I

Multpliziere (6) mit 2, (1) mit 3 und addiere die resultierenden Ungleichungen

=> $-6x_1 + 8x_2 \leq 20$

$6x_1 + 9x_2 \leq 81$

=> $17 x_2 \leq 101$

=> gewünschte Ungleichung $x_2 \leq \lfloor 101/17 \rfloor = 5$

Allgemein haben diese Unleichungen die Form

$y^T Ax \leq \lfloor y^T b \rfloor$ mit $y \geq 0$ und $y^T A$ ganzzahlig

wobei $Ax \leq b$ das System der Ungleichungen nach dem "vorigen" Schritt ist.

Aus dieser Beobachtung leitet sich die allgemeine Definition eines Schnittebenenbeweises ab.

Allgemeine Definition

Sei $Ax \leq b$ ein Ungleichungssystem mit m Ungleichungen.

Ein Schnittebenenbeweis für die Ungleichung $w^T x \le t$ mit ganzzahligem w und t ausgehend von $Ax \le b$ ist eine endliche Folge von Ungleichungen der Form

$$a_{m+k}^T x \le b_{m+k} \quad (k = 1, \ldots, M)$$

zusammen mit nichtnegativen Zahlen

$$y_{k_j} \quad (1 \le k \le M,\ 1 \le j \le m + k - 1)$$

so dass für jedes $k = 1, \ldots, M$ die Ungleichung

$$a_{m+k}^T x \le b_{m+k}$$

sich als nichtnegative Linearkombination

$$(y_{k_1}, \ldots, y_{k_{m+k-1}}) \begin{pmatrix} a_{11}x_1 + \ldots + a_{1n}x_n \\ \vdots \\ a_{m+k-1,1}x_1 + \ldots + a_{m+k-1,n}x_n \end{pmatrix} \le \lfloor (y_{k_1}, \ldots, y_{k_{m+k-1}}) \begin{pmatrix} b_1 \\ \vdots \\ b_{m+k-1} \end{pmatrix} \rfloor$$

der bereits vorhandenen Ungleichungen ergibt, wobei

- die Koeffizienten der Variablen auf der linken Seite ganzzahlig sind
- die rechte Seite nicht ganzzahlig ist und nach unten gerundet wird, und
- die letzte Ungleichung der Folge die Ungleichung $w^T x \le t$ ist

Ungleichungen dieser Form

$$y^T A x \le \lfloor y^T b \rfloor \quad \text{mit } y \ge 0 \text{ und } y^T A \text{ ganzzahlig}$$

heißen Gomory-Chvátal Schnitte.

Gomory hat 1960 gezeigt, dass man mit diesen Schnitten ein endliches Schnittebenenverfahren erhält.

Chvátal hat 1973 das hier behandelte Prinzip der Schnittebenenbeweise eingeführt. Diese Beweise ähneln dem Farkas Lemma in den Varianten Lemma 7.21 (2) und 7.22.

7.24 Satz (Schnittebenenbeweise für rationale Polytope, Chvátal 1973)

Sei $P = \{ x \in \mathbb{R}^n \mid Ax \le b \}$ ein rationales Polytop und sei $w^T x \le t$ eine Ungleichung mit ganzzahligem w und t, die von allen Punkten aus P_I erfüllt wird. Dann gibt es einen Schnittebenenbeweis von $Ax \le b$ für eine Schnittebene $w^T x \le t'$ mit $t' \le t$.

Beweis siehe unten ❑

7.25 Satz (Schnittebenenbeweise für rationale Polytope ohne ganzzahlige Punkte, Chvátal 1973)

Sei $P = \{ x \in \mathbb{R}^n \mid Ax \le b \}$ ein rationales Polytop ohne ganzzahlige Punkte. Dann existiert ein Schnittebenenbeweis von $Ax \le b$ für die Ungleichung $0^T x \le -1$.

Beweis siehe unten ❑

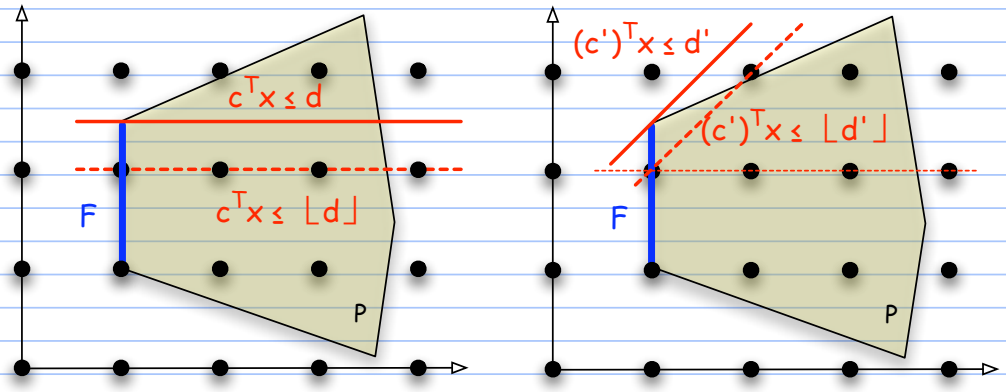
Für die Beweise benötigen wir ein Lemma, das eine Induktion über die Dimension von P möglich macht. Es zeigt, dass Gomory-Chvátal Schnitte von Seitenflächen eines rationalen Polyeders auf das Polyeder selbst "durch Rotation" geliftet werden können.

7.26 Lemma (Rotation von Gomory-Chvátal Schnitten)

Sei F eine durch ein Gleichungssystem beschriebene Seitenfläche eines rationalen Polytops P und sei $c^T x \le \lfloor d \rfloor$ ein Gomory-Chvátal Schnitt für F.

Dann existiert ein Gomory-Chvátal Schnitt $(c')^T x \le \lfloor d' \rfloor$ für P mit

$$F \cap \{ x \mid c^T x \le \lfloor d \rfloor \} = F \cap \{ x \mid (c')^T x \le \lfloor d' \rfloor \} \text{ (Gleichheit auf } F)$$



Beweis

Sei o.B.d.A. $P = \{ x \mid A'x \le b',\ A''x \le b'' \}$ mit A'', b'' ganzzahlig.

Sei $F := \{ x \mid A'x \le b',\ A''x = b'' \}$ (Gleichungen $A''x = b''$ beschreiben F)

Sei $c^T x \le \lfloor d \rfloor$ der Schnitt für F laut Voraussetzung

und o.B.d.A. $d = \max \{ c^T x \mid x \in F \}$ (schärfster Schnitt mit $c^T x$; existiert da P ein Polytop).

Dualitätzsatz => das duale LP hat eine Optimallösung

=> es gibt $y' \ge 0$ und y'' beliebig mit

$(y')^T A' + (y'')^T A'' = c^T$ (*)

$(y')^T b' + (y'')^T b'' = d$ (**)

Konstruktion von c' und d' aus y''

$(c')^T := c^T - (\lfloor y'' \rfloor)^T A'' = (y')^T A' + (y'' - \lfloor y'' \rfloor)^T A''$ wegen (*)

ganzzahlig ≥ 0 ≥ 0

$d' := d - (\lfloor y'' \rfloor)^T b'' = (y')^T b' + (y'' - \lfloor y'' \rfloor)^T b''$ wegen (**)

c ganzzahlig als Gomory-Chvátal Schnitt, $(\lfloor y'' \rfloor)^T A''$ ganzzahlig $\Rightarrow$ c' ganzzahlig

$(c')^T x \leq d'$ ist gültige Ungleichung für P

denn $(c')^T x = (y')^T A' x + (y'' - \lfloor y'' \rfloor)^T A'' x \leq (y')^T b' + (y'' - \lfloor y'' \rfloor)^T b'' = d'$

≥ 0 ≥ 0

Definition von d' $\Rightarrow$ $d = d' + (\lfloor y'' \rfloor)^T b''$

ganzzahlig, da b'' ganzzahlig

$\Rightarrow \lfloor d \rfloor = \lfloor d' \rfloor + (\lfloor y'' \rfloor)^T b''$

Gleichheit auf F

$F \cap \{ x \mid (c')^T x \leq \lfloor d' \rfloor \}$

$= F \cap \{ x \mid (c')^T x \leq \lfloor d' \rfloor, (\lfloor y'' \rfloor)^T A'' x = (\lfloor y'' \rfloor)^T b'' \}$

erfüllt in F wegen $A''x = b''$

$= F \cap \{ x \mid (c' + (\lfloor y'' \rfloor)^T A'')^T x \leq (\lfloor d' \rfloor + \lfloor y'' \rfloor)^T b'' \}$

$= F \cap \{ x \mid c^T x \leq \lfloor d \rfloor \}$ ❑

Beweis von Satz 7.25 (Schnittebenenbeweise für rationale Polytope ohne ganzzahlige Punkte)

Induktion nach dim(P)

Induktionsanfang

$P = \emptyset$

$\Rightarrow$ Behauptung mit Farkas Lemma 7.21 (3)

dim(P) = 0

$\Rightarrow$ $P = \{x^*\}$ und x^* ist nicht ganzzahlig.

$\Rightarrow$ (Induktion nach n) es gibt gzz. Vektor w mit $w^T x^*$ nicht ganzzahlig

Sei t so, dass die Hyperebene $H = \{ x \mid w^T x = t \}$ durch x^* geht (durch Verschiebung immer erreichbar).

$w^T x^*$ nicht ganzzahlig $\Rightarrow$ $t = w^T x^*$ nicht ganzzahlig

$\Rightarrow$ $w^T x \leq t$ ist gültig für P, aber $P' := P \cap \{ x \mid w^T x \leq \lfloor t \rfloor \} = \emptyset$

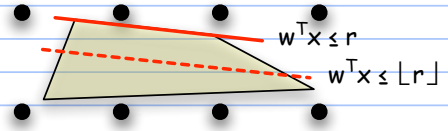
Farkas Lemma 7.22 => es gibt Schnittebenenbeweis für $w^Tx \le \lfloor t \rfloor$ von $Ax \le b$

$P' = \emptyset$ => (Farkas Lemma 7.21 (3)) es gibt Schnittebenenbeweis für $0^Tx \le -1$ von $Ax \le b$ and $w^Tx \le \lfloor t \rfloor$

Induktionsschluss auf $\dim(P) \ge 1$

Sei $w^Tx \le r$, w ganzzahlig, eine Ungleichung, die eine echte Seitenfläche von P erzeugt.

Sei $P' := \{ x \in P \mid w^Tx \le \lfloor r \rfloor \}$



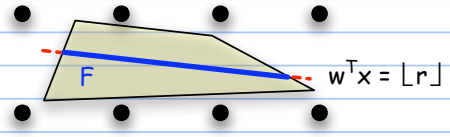
Fall 1: $P' = \emptyset$

Farkas Lemma 7.22 => können $w^Tx \le r$ aus $Ax \le b$ herleiten

Farkas Lemma 7.21 => können $0^Tx \le -1$ aus $Ax \le b, w^Tx \le \lfloor r \rfloor$ herleiten

Fall 2: $P' \neq \emptyset$

Sei $F := \{ x \in P' \mid w^Tx = \lfloor r \rfloor \}$ => F ist Seitenfläche von P'



Claim: $\dim(F) < \dim(P)$

denn:

entweder F ist echte Seitenfläche von P (wenn r ganzzahlig)

oder $w^Tx \le \lfloor r \rfloor$ schneidet von P etwas ab, dann enthält P Punkte, die nicht $w^Tx = \lfloor r \rfloor$ erfüllen

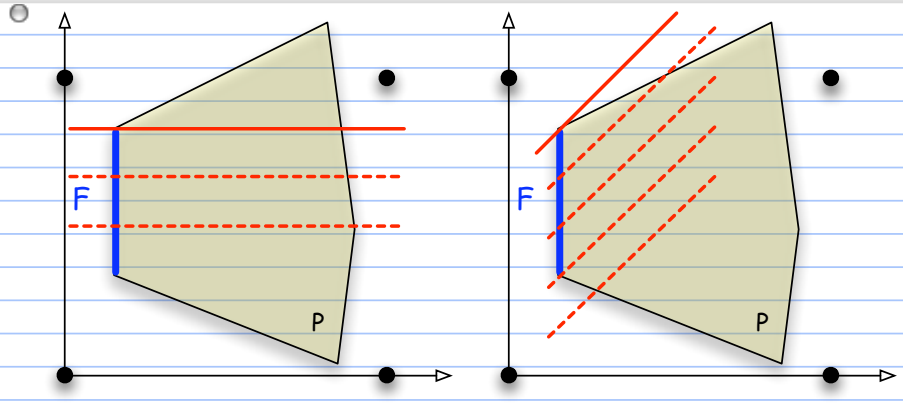
=> niedrigere Dimension in beiden Fällen

F_I leer, Induktionsvoraussetzung

=> es gibt einen Schnittebenenbeweis für $0^Tx \le -1$ von $Ax \le b, w^Tx \le \lfloor r \rfloor, -w^Tx \le -\lfloor r \rfloor$

Neue erhaltene Schnittebene $0^Tx \le -1$ bzgl. F mehrfach nutzen und jedesmal Rotationslemma anwenden

ergibt folgendes Bild



P Polytop => min { $w^T x \mid x \in P$ } ist endlich =>

Wiederholung des Arguments mit $P'' := \{ x \in P \mid w^T x \leq \lfloor r \rfloor - 1 \}$ usw. ergibt nach endlich vielen Schritten einen Schnittebenenbeweis für eine Ungleichung $w^T x \leq t$ mit $\{ x \in P \mid w^T x \leq t \} = \emptyset$

=> Reduktion auf Fall 1 ❑

Beweis von Satz 7.24 (Schnittebenenbeweise für rationale Polytope)

Fall 1: $P_I = \emptyset$

Satz 7.25 => $0^T x \leq -1$ ist beweisbar

P Polytop => $r := \max \{ w^T x \mid x \in P \}$ ist endlich

=> $w^T x \leq r$ ist gültige Ungleichung für P

Farkas Lemma 7.22 => $w^T x \leq r$ ist beweisbar

w ganzzahlig => $w^T x \leq \lfloor r \rfloor$ ist Gomory-Chvátal Schnitt

Addition von $w^T x \leq \lfloor r \rfloor$ und $0^T x \leq -1$ ergibt $w^T x \leq \lfloor r \rfloor - 1$

Fortgesetzte Addition von $0^T x \leq -1$ ergibt $w^T x \leq t' \leq t$ in endlich vielen Schritten

Fall 2: $P_I \neq \emptyset$

P Polytop => $r := \max \{ w^T x \mid x \in P \}$ ist endlich

Sei $P' := \{ x \in P \mid w^T x \leq \lfloor r \rfloor \}$

Falls $\lfloor r \rfloor \leq t$ => fertig

Sei also $\lfloor r \rfloor > t$

Sei $F := \{ x \in P' \mid w^T x = \lfloor r \rfloor \}$ => F ist Seitenfläche von P'

F enthält keine ganzzahligen Punkte, da $w^T x \leq t$ gültig für P_I und $t < \lfloor r \rfloor$ ist

Theorem 7.25 => für F gibt es einen Schnittebenenbeweis von $0^T x \leq -1$ ausgehend von $Ax \leq b, w^T x = \lfloor r \rfloor$

Rotations Lemma für F and P' => es gibt einen Schnittebenenbeweis $c^T x \leq \lfloor d \rfloor$ für P ausgehend von $Ax \leq b, w^T x \leq \lfloor r \rfloor$ so dass

$P' \cap \{ x \mid c^T x \le \lfloor d \rfloor , w^T x = \lfloor r \rfloor \} = \emptyset$

- Die Anwendung dieser Schnitte auf P' ergibt den Schnitt $w^T x \le \lfloor r \rfloor - 1$.
- Wiederholung des Arguments ergibt irgendwann $w^T x \le t' \le t$ und führt zu Fall 1 ❑

Chvátal Hülle und Chvátal Rank

Schnittebenenbeweise benutzen bereits erzeugte Schnittebenen im Beweis. Wir betrachten jetzt, was geschieht, wenn man nur die ursprünglich gegebenen Schnittebenen $Ax \le b$ benutzen darf

Sei $P = \{ x \in \mathbb{R}^n \mid Ax \le b \}$ ein rationales Polytop. Fügt man alle Gomory-Chvátal Schnitte $y^T Ax \le \lfloor y^T b \rfloor$ mit $y \ge 0$, $y^T A$ ganzzahlig zu P hinzu, so erhält man die Chvatal Hülle P' von P.

7.27 (Eigenschaften der Chvatal Hülle)

Die Chvátal Hülle eines rationalen Polytopes ist wieder ein rationales Polytop. Insbesondere braucht man zu seiner linearen Beschreibung nur $Ax \le b$ und endlich viele der Gomory-Chvátal Schnitte.

Beweis

Sei $P = \{ x \mid Ax \le b \}$ mit A und b ganzzahlig

Setze $P' := P \cap \{ x \mid y^T Ax \le \lfloor y^T b \rfloor$ mit $y \ge 0$, $y^T A$ ganzzahlig $\}$. Es gilt:

(7.13) $P' = P \cap \{ x \mid y^T Ax \le \lfloor y^T b \rfloor$ mit $y \ge 0$, $y^T A$ ganzzahlig, $0 \le y < 1 \}$

Beweis (7.13):

Sei $w^T x \le \lfloor t \rfloor$ ein Gomory-Chvátal Schnitt mit $y \ge 0$, $y^T A = w$, $y^T b = t$

Sei $y' := y - \lfloor y \rfloor$ der fraktionale Teil von y und $0 \le y' < 1$

Sei $w' := (y')^T A = y^T A - (\lfloor y \rfloor)^T A = w - (\lfloor y \rfloor)^T A$

=> w' ist ganzzahlig, da w und A ganzzahlig sind

Sei $t' := (y')^T b = y^T b - (\lfloor y \rfloor)^T b = t - (\lfloor y \rfloor)^T b$

=> t und t' unterscheiden sich um eine ganze Zahl, nämlich $(\lfloor y \rfloor)^T b$

=> $w^T x \le \lfloor t \rfloor$ ergibt sich als Summe aus

$(w')^T x \le \lfloor t' \rfloor$ <-- gemäß (7.13) gebildet

$+ (\lfloor y \rfloor)^T Ax \le (\lfloor y \rfloor)^T b$ <-- redundant, da

nichtnegative Linearkombination der Zeilen von $Ax \le b$

=> die Ungleichungen gemäß (7.13) reichen zur Bildung der Hülle

Es gibt nur endlich viele Ungleichungen gemäß (7.13)

Seien a_{ij} die Einträge der Matrix A und sei A_j die j-te Spalte von A

=> $y^T A_j \in [\, -\sum_i |a_{ij}|, \sum_i |a_{ij}| \,]$ und ganzzahlig für $0 \le y < 1$

=> es gibt nur endlich viele $y^T A_j$, die dies leisten

Alle Ungleichungen der Form (7.13) haben ganzzahlige Koeffizienten

=> sie sind wieder rational ❑

Die Chvátal Hüllenbildung lässt sich natürlich iterieren und liefert eine Folge

$P = P^{(0)} \supseteq P^{(1)} \supseteq P^{(2)} \supseteq \ldots \supseteq P_I$

7.28 Satz (Die Hüllenbildung ist endlich)

Sei P ein rationales Polytop. Dann existiert ein $k \in \mathbb{N}$ mit $P_I = P^{(k)}$.

Insbesondere terminiert das Schnittebenenverfahren bei geigneter Auswahl von Gomory Chvátal Schnitten nach endlich vielen Schritten.

Beweis

P_I ist ein Polytop und damit durch endlich viele Ungleichungen beschreibbar.

Für jede dieser Ungleichungen existiert ein Schnittebenenbeweis einer endlichen Länge r,

also mit Ungleichungen nur aus endlich vielen der $P^{(i)}$

=> das Maximum dieser r erfüllt die Behauptung ❑

Gomory hat 1960 eine solche geeignete Auswahl beschrieben.

Das kleinste k mit $P_I = P^{(k)}$ heißt der Chvátal Rang von P. Dieser bildet eine Art Komplexitätsmaß für die ganzzahlige Hülle von Polytopen. Er kann bereits für Polytope im $\mathbb{R}^2$ beliebig groß werden, ist jedoch für Polytope im 0/1-Würfel im $\mathbb{R}^n$ durch $6n^3 \log n$ beschränkt.

7.29 Beispiel (Ein Polytop mit Chvátal Rang 2)

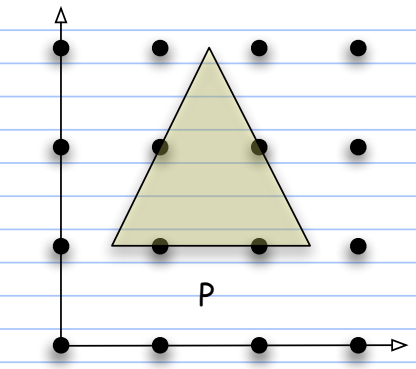
Das Ausgangspolytop P

P sei gegeben durch

$-2x_1 + x_2 \le 0 \quad (1)$

$2x_1 + x_2 \le 6 \quad (2)$

$-x_2 \le -1 \quad (3)$



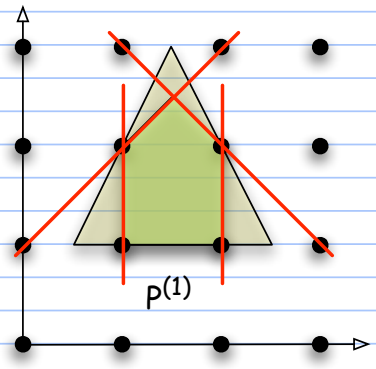
Die erste Chvatal Hülle $P^{(1)}$

$y^T = (0, 1/2, 1/2)$ ergibt $x_1 \leq 5/2 \Rightarrow x_1 \leq 2$

$y^T = (1/2, 0, 1/2)$ ergibt $-x_1 \leq -1/2 \Rightarrow -x_1 \leq -1$

$y^T = (5/6, 1/3, 1/6)$ ergibt $-x_1 + x_2 \leq 11/6 \Rightarrow -x_1 + x_2 \leq 1$

$y^T = (1/3, 5/6, 1/6)$ ergibt $x_1 + x_2 \leq 29/6 \Rightarrow x_1 + x_2 \leq 4$



$x_2 \leq 2$ ist nicht aus $Ax \leq b$ ableitbar

allgemeiner Gomory-Chvátal Schnitt ist

$(-2y_1 + 2y_2)x_1 + (y_1 + y_2 - y_3)x_2 \leq \lfloor 6y_2 - y_3 \rfloor$

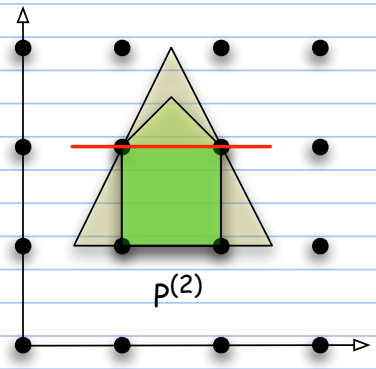
$\Rightarrow -2y_1 + 2y_2 = 0,\ y_1 + y_2 - y_3 = 1,\ \lfloor 6y_2 - y_3 \rfloor = 2$

$\Rightarrow y_1 = y_2,\ y_3 = 2y_2 - 1,\ \lfloor 4y_2 + 1 \rfloor = 2 \Leftrightarrow 1 \leq 4y_2 < 2$

$\Rightarrow y_2 < 1/2 \Rightarrow y_3 < 0 \Rightarrow$ Widerspruch

Die zweite Chvatal Hülle $P^{(2)}$

$x_2 \leq 2$ ist ableitbar aus $-x_1 + x_2 \leq 1,\ x_1 + x_2 \leq 4$ mit $y^T = (1/2, 1/2)$



Abschließende Bemerkungen

- Alle hier getroffenen Ausagen über Schnittebenenbeweise und Chvátal Rang gelten auch für beliebige rationale Polyeder (vgl. Korte & Vygen), aber i.A. nicht mehr für nicht-rationale Polyeder.
- Gomory-Chvátal Schnitte sind ein Standardwerzeug in CPLEX. Eine Auswahl wird bei ganzzahligen Programmen automatisch erzeugt und bereits der LP-Relaxation hinzugefügt. Im Laufe des Branch & Bound Algorithmus werden dann weitere speziell für die Teilprobleme in Knoten des Branch & Bound Baumes erzeugt.
- Für viele kombinatorische Optimierungsprobleme gibt es Aussagen zum Chvátal Rang bestimmter Ungleichungen:
  - Die Odd-Set Ungleichungen des Matching Polytops haben Chvátal Rang 1 bzgl. der LP Formulierung, die nur die

Gradbedingungen enthält.

$x(\delta(v)) = 1$ für alle $v \in V$ Gradbedingungen

$\sum_{e \subseteq R} x_e \leq r$ für alle Mengen $R \subseteq V(G)$ mit $|R| = 2r+1$ Odd-Set Ungleichungen

$x \geq 0$

- Die Kamm-Ungleichungen für das TSP haben Chvátal Rang 1 bzgl. des 2-Matching Polytops (siehe Abschnitt 8.2)

- Separierung ist das Problem, zu einem gegebenem Punkt x^* und einem Polyeder Q eine Hyperebene H zu ...
- Optimierung (OPT)
  - Input:
    - rationales Polyeder Q,
      $c \in \mathbb{R}^n$ so dass $c^T x$ ist auf P nach unten beschränkt ist
  - Output:
    - $x^* \in Q$ mit $x^* = \min \{ c^T x \mid x \in Q \}$
- Separierung (SEP)
  - Input:
    - rationales Polyeder Q,
      $y \in \mathbb{R}^n$
  - Output:
    - "Ja" falls $y \in Q$
      $d \in \mathbb{R}^n$ mit $d^T x < d^T y$ für alle $x \in Q$ falls $y \notin Q$ (eine separierende Hyperebene)

- 7.30 Satz (Polynomiale Äquivalenz von Separierung und Optimierung; Grötschel, Lovasz, Schrijver 1984)

- (OPT) polynomial lösbar <=> (SEP) polynomial lösbar
  Dies gilt auch für ε-Approximationen
- ohne Beweis,
  bei volldimensionalen Polyedern werden folgende Techniken benutzt
  "<=" Ellipsoidmethode und Dualitätssatz
  "=>" Antiblocking von Polyedern
  Details siehe
  M. Grötschel, L. Lovász, and A. Schrijver,
  Geometric Algorithms and Combinatorial Optimization,
  Springer-Verlag, Berlin, 2nd ed., 1993. ❑

- Bemerkungen
  - Ist (OPT) polynomial lösbar, so sind Schnittebenen effizient ermittelbar
  - Ist (OPT) NP-schwer, so wird man (falls P ≠ NP) nicht alle Schnittebenen in polynomialer Zeit finden, aber unter Umständen doch noch viele.
  - Daher verwendet man polynomiale Algorithmen zur Ermittlung von Schnittebenen, bis diese keine mehr finden

und brancht dann nach fraktionalen Variablen. Für die entstehenden Unterprobleme sucht man wieder Schnittebenen bis man brachen muss usw.

Diese Kombination von Branch & Bound mit Schnittebenenverfahren nennt man Branch & Cut. Beispiele siehe Kapitel 8.

- Statt der Ellipsoidmethode (die sich als ineffizient für die Praxis erwiesen hat) verwendet man üblicherweise den dualen Simplexalgorithmus, der es einfach gestattet, neue Schnittebenen als zusätzliche Restriktionen einzubauen.

8. Von Kombinatorischen Optimierungsproblemen induzierte Polytope 48

- Ziel dieses Abschnittes
  - Abstrakte Sicht auf kombinatorische Optimierungsprobleme entwickeln um induzierte Polytope einheitlich beschreiben zu können.

- Instanz eines (abstrakten) Kombinatorischen Optimierungsproblems ist ein Tripel $(E, \mathcal{F}, c)$ mit
  - E ist eine endliche Menge, die Grundmenge (z.B. die Menge der Kanten eines Graphen)
  - $\mathcal{F}$ ist das Mengensystem der zulässigen Lösungen $F \subseteq E$ (z.B. das Mengensystem aller Matchings $M \subseteq E(G)$)
  - $c : E \to \mathbb{R}$ mit $c(F) := \sum_{e \in F} c(e)$ gibt den Wert der zulässigen Lösung F an (z.B. das Gewicht eines Matching M)

- Das von $(E, \mathcal{F}, c)$ induzierte Polytop $P_{\mathcal{F}}$ ergibt sich folgendermaßen
  - für eine Menge $F \subseteq E$ betrachten wir den Inzidenzvektor $x^F \in \mathbb{R}^E$ mit

$$x_e^F := \begin{cases} 1 & e \in F \\ 0 & e \notin F \end{cases}$$

  ferner interpretieren wir c als Vektor $c \in \mathbb{R}^E$ mit $c_e := c(e)$

  und definieren für beliebige Vektoren $x \in \mathbb{R}^E$

$$x(F) := \sum_{e \in F} x_e$$

  - dann ist $P_{\mathcal{F}} := \text{conv}\{ x^F \mid F \in \mathcal{F} \}$,

    d.h. gleich der konvexen Hülle aller Inzidenzvektoren x^F von zulässigen Lösungen $F \in \mathcal{F}$
- Nach dem Satz von Minkowski gilt ...
- Probleme:
  - Wie findet man lineare Beschreibungen?

    Der Beweis des Satzes von Minkowski ist konstruktiv, aber zu "unhandlich".
  - Wie groß ist die Anzahl der Restriktionen?

    i.A. exponentiell, vgl. Matching Polytop,

    es reicht jedoch eine partielle Beschreibung für die interessierende Ecke
- Mit diesen Problemen beschäftigt sich die kombinatorische Polyedertheorie (polyhedral combinatorics)

- Ziel dieses Abschnittes
  - Illustration einiger (partieller) linearer Beschreibungen und der Technik, wie man sie beweist

- **Das Matching Polytop**
  - hier ist $(E, \mathcal{F}, c)$ gegeben durch

    E = Kantenmenge $E(G)$ eines ungerichteten Graphen

    $\mathcal{F} = \{ M \subseteq E \mid M \text{ ist ein Matching} \} =: \mathcal{M}$

    $c(e)$ = nichtnegative Kantenbewertung von e

  $P_{\mathcal{M}} := \text{conv} \{ x^M \mid M \in \mathcal{M} \}$ heißt Matching Polytop des Graphen G

- 8.1 Satz (lineare Beschreibung des Matching Polytops)
  - In bipartiten Graphen hat $P_{\mathcal{M}}$ die lineare Beschreibung

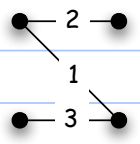
    $x(\delta(v)) \leq 1$ für alle Knoten v

    $x \geq 0$
  - In beliebigen Graphen hat $P_{\mathcal{M}}$ die lineare Beschreibung

    $x(\delta(v)) \leq 1$ für alle Knoten v

    $\sum_{e \subseteq R} x_e \leq r$ für alle ungeraden Mengen $R \subseteq V(G)$ mit $|R| = 2r+1$

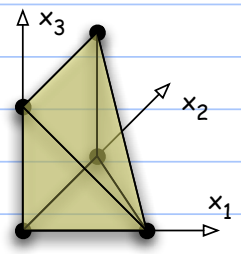
    $x \geq 0$
  - Beispiel
    - G bipartit



    - $\mathcal{M} = \{ \emptyset, \{1\}, \{2\}, \{3\}, \{2,3\} \}$

      Variable x_1, x_2, x_3 für die Kanten 1, 2, 3

      dann ergibt sich $P_{\mathcal{M}} := \text{conv} \{ x^M \mid M \in \mathcal{M} \}$ als eine gekippte Pyramide mit quadratischer Grundfläche in der x_2-x_3 Ebene



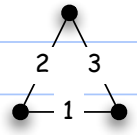
- Als lineare Beschreibung von $P_{\mathcal{M}}$ erhält man (redundante Ungleichungen für Knoten vom Grad 1 weggelassen)

  $x_1 + x_2 \leq 1$

  $x_1 + x_3 \leq 1$

  $x_i \geq 0$

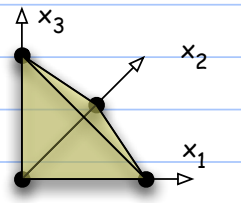
- *G* nicht bipartit



- $\mathcal{M} = \{ \emptyset, \{1\}, \{2\}, \{3\} \}$

  Variable x_1, x_2, x_3 für die Kanten 1, 2, 3

  dann ergibt sich $P_{\mathcal{M}} := \text{conv} \{ x^M \mid M \in \mathcal{M} \}$ als ein Tetraeder



- Die Ungleichungen für den bipartiten Fall werden von $x := (1/2, 1/2, 1/2)^T$ erfüllt, aber $x \notin P_{\mathcal{M}}$

  Daher braucht man die zusätzlichen Ungleichungen für ungerade Mengen, hier also

  $x_1 + x_2 + x_3 \leq 1$

Der Beweis beruht in beiden Fällen auf folgenden Ideen

- (1) die Ungleichungen sind gültig für $P_{\mathcal{M}}$, d.h. $P_{\mathcal{M}} \subseteq \{ x \mid Ax \leq b, x \geq 0 \}$
- (2) Bzgl. der Optimierung

  $\max c^T x$ über den Ungleichungen $Ax \leq b, x \geq 0$

wird das Optimum immer in einem Inzidenzvektor x^M, $M \in \mathcal{M}$ angenommen

(z.B. gezeigt durch Satz vom komplementären Schlupf oder primal-dualem Algorithmus)

(3) Lemma 3.5 (zu jeder Ecke x gibt es eine Zielfunktion c, so dass das Optimum genau in x angenommen wird)

(2), (3) => jede Ecke der linearen Beschreibung ist Inzidenzvektor x^M, $M \in \mathcal{M}$

=> $\{ x \mid Ax \leq b, x \geq 0 \} \subseteq P_{\mathcal{M}}$

Das Polytop der Antiketten einer partiellen Ordnung

Grundbegriffe über partielle Ordnungen

Eine (endliche) partielle Ordnung $(E, <)$ ist gegeben durch

eine (endliche) Grundmenge E und

eine binäre Relation $<$ auf E mit

$a < b$ und $b < c$ => $a < c$ ($<$ ist transitiv)

$a < b$ => $a \neq b$ ($<$ ist irreflexiv)

Wir stellen partielle Ordnungen dar durch ein Kantendiagramm

= azyklischer Digraph $G = (V, E)$ mit

Kanten = Grundmenge der partiellen Ordnung

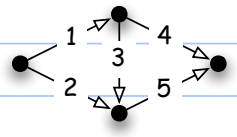
$e < e'$:<=> es gibt gerichteten Weg von head(e) nach tail(e')

head(e) = head(e') für alle minimalen Elemente e, e' der partiellen Ordnung $(E, <)$

tail(e) = tail(e') für alle maximalen Elemente e, e' der partiellen Ordnung $(E, <)$

(jede partielle Ordnung ist als induzierte Subordnung eines Kantendiagramms darstellbar)

Beispiel:



$E = \{ 1, 2, 3, 4, 5 \}$ und $1 < 4$, $1 < 3 < 5$, $2 < 5$

$a, b \in E$ heißen vergleichbar :<=> $a < b$ oder $b < a$

Kette = Menge paarweise vergleichbarer Elemente

im Beispiel etwa $\emptyset$ {3} {1, 5} {1,4} {1,3,5}, die letzten beiden sind maximale Ketten (d.h. $\subseteq$-maximal)

$a, b \in E$ heißen unvergleichbar :<=> a, b sind nicht vergleichbar

Antikette = Menge paarweise unvergleichbarer Elemente

im Beispiel etwa $\emptyset$ {3} {2, 3} {4,5} {2,3,4}, die letzten beiden sind maximale Antiketten (d.h. $\subseteq$-maximal)

Kombinatorisches Optimierungsproblem

Bestimme eine Antikette maximalen Gewichts bzgl. Gewichten $c(e) \geq 0$

Anwendung: Projektscheduling

partielle Ordnung = Ablaufstruktur eines Bauprojektes

Kette = was sequentiell gemacht werden muss

Antikette = was parallel durchgeführt werden kann

Gewicht einer Antikette = Bedarf an Betriebsmitten, wenn sie parallel dürchgeführt wird

maximales Gewicht = maximaler Betriebsmittelbedarf

hier ist $(E, \mathcal{F}, c)$ gegeben durch

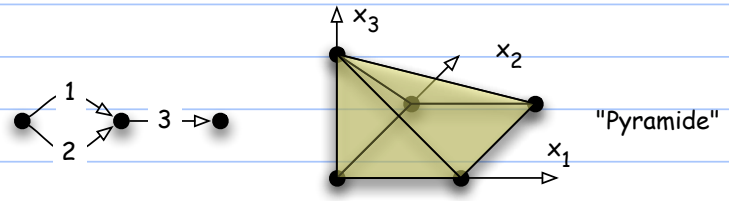
E = Kantenmenge $E(G)$ des Kantendiagramms der partiellen Ordnung

$\mathcal{F} = \{ A \subseteq E \mid A \text{ ist Antikette} \} =: \mathcal{A}$

$c(e)$ = nichtnegative Kantenbewertung von e

$P_{\mathcal{A}} := \text{conv}\{ x^A \mid A \in \mathcal{A} \}$ heißt Antiketten Polytop der partiellen Ordnung

Beispiel



"Pyramide"

Lineare Beschreibung von $P_{\mathcal{A}}$

Notwendige Bedingung: jede Kette enthält von einer Antikette höchstens ein Element.

Als Ungleichung formulieren: $x(K) \leq 1$ für jede Kette K

8.2 Satz (lineare Beschreibung des Antikettenpolytops)

$P_{\mathcal{A}}$ hat die lineare Beschreibung

$x(K) \leq 1$ für jede Kette K

$x \geq 0$

Beweis

(1) alle Ungleichungen sind gültig für das Antikettenpolytop

Sei $x \in P_{\mathcal{A}}$ und K eine Kette

=> x ist Konvexkombination von Ecken von $P_{\mathcal{A}}$, also von Inzidenzvektoren x^A von Antiketten A

=> jeder Inzidenzvektor x^A erfüllt $x^A(K) \leq 1$

=> die Konvexkombination x erfüllt $x(K) \leq 1$

(2) Das LP über den Ungleichungen nimmt das Optimum auf Inzidenzvektoren von Antiketten an

Das LP ist gegeben durch

(P) max $c^T x$ unter $x(K) \leq 1$ für jede Kette K

$x \geq 0$

Das duale LP ist

(D) min $\mathbf{1}^T y = \Sigma_K y_K$ unter $\Sigma_{K: e \in K} y_K \geq c_e$ für alle $e \in E$

$y_K \geq 0$ für alle Ketten K

Falls y_K als "Vielfachheit" der Kette K interpretiert wird, so sagt (D):

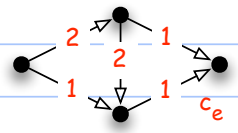
finde möglichst wenige Ketten (gleiche mehrmals zugelassen), so dass jedes Element e mindestens c_e mal "überdeckt" wird

=> Beschränkung auf maximale Ketten sinnvoll

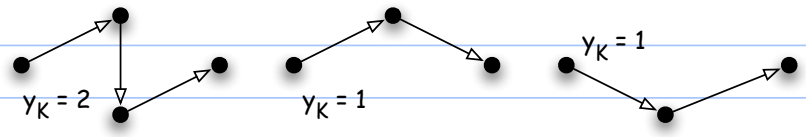
=> $\Sigma_K y_K$ ist Fluss im Kantendiagramm G bzgl. unterer Kapazitäten c_e

Beispiel:

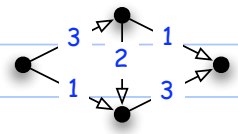
G mit unteren Kapazitäten



die Überdeckung mit Ketten



die Interpretation von $\Sigma_K y_K$ als Fluss



=> der Min-Fluss-Max-Schnitt Satz ist anwendbar

- analog zum Max-Fluss-Min-Schnitt Satz der ADM I:
- Der minimale Flusswert in einem s,t-Netzwerk mit unteren Kapazitäten c_e und oberen Kapazitäten u_e ist gleich der maximalen Kapazität cap(X) eines s,t-Schnittes X
- dabei ist Kapazität eines s,t-Schnittes X mit $s \in X$ definiert als

$$\text{cap}(X) = \sum_{e \in \delta^+(X)} c_e - \sum_{e \in \delta^-(X)} u_e$$

- hier sind alle oberen Kapazitäten ∞ und der Flusswert ist nach unten beschränkt
  => die maximale Kapazität eines Schnittes hat endlichen Wert
  => der Schnitt maximaler Kapazität besteht nur aus Vorwärtskanten, d.h. $\delta^-(X) = \emptyset$
  => $A := \delta^+(X)$ ist eine Antikette (sonst ist $\delta^-(X) \neq \emptyset$), deren Gewicht gleich der maximalen Kapazität eines Schnittes ist
- umgekehrt definiert jede maximale Antikette A einen Schnitt X mit
  X := Menge der Anfangsknoten von Kanten in A und von Knoten aller Kanten e mit $e < a$ für ein $a \in X$
  und es gilt $A := \delta^+(X)$
- Dann folgt:
  Optimalwert von (P) = Optimalwert von (D)

= minimaler Flusswert = maximale Kapazität eines s,t-Schnittes
= maximales Gewicht einer Antikette
=> Optimalwert von (P) wird auf Inzidenzvektoren von Antiketten angenommen

- (1) und (2) ergeben die Behauptung mit den Vorüberlegungen zum Beweisprinzip beim Matchingpolytop ❑

8.3 Bemerkung

- (P) und (D) entsprechen einer gewichteten Form des Satzes von Dilworth
- Satz von Dilworth:
  - minimale Anzahl von Ketten, die alle Elemente einer partiellen Ordnung überdecken
    = maximale Kardinalität einer Antikette

Das Traveling Salesman Polytop im symmetrischen, vollständigen Fall

Betrachte das TSP auf einem ungerichtenen, o.B.d.A. vollständigen Graphen K_n mit Kantenkosten $c(e) \geq 0$.
Hier ist $(E, \mathcal{T}, c)$ gegeben durch

E = Kantenmenge $E(K_n)$
$\mathcal{T}$ = { $T \subseteq E$ | T ist TSP-Tour }

c(e) = nichtnegative Kantenbewertung von e

$P_{\mathcal{T}} := \text{conv}\{ x^T \mid T \text{ ist TSP-Tour} \}$ heißt TSP Polytop und wird mit Q^n_{TSP} bezeichnet

- In den Knoten muss gelten

  $x(\delta(v)) = 2$ für alle Knoten v (8.1)

  => $\dim(Q^n_{TSP}) \leq m - n = \frac{n(n-3)}{2}$

  Es gilt sogar $\dim(Q^n_{TSP}) = m - n$

  => (8.1) definiert ein Gleichungssystem maximalen Rangs, dessen Lösungsmenge Q^n_{TSP} enthält

- Die Bedingungen (8.1) lassen noch die Vereinigung von Subtouren zu.

  Um diese zu vermeiden, führt man die Subtour-Eliminationsbedingungen ein:

  $x(E(W)) \leq |W| - 1$ für alle $\emptyset \neq W \subset V$ (8.2)

  dabei ist E(W) = Menge der Kanten mit beiden Endpunkten in W

- Ferner muss natürlich gelten (Vorzeichenbedingung, Beschränkung)

$0 \leq x_e \leq 1$ für alle Kanten e (8.3)

- Q^n_{TSP} ist in folgenden Polytopen enthalten (Relaxationen von Q^n_{TSP})
  - im 1-Baum Polytop (= konvexe Hülle der Inzidenzvektoren von 1-Bäumen) vgl. Abschnitt 7.4
  - im 2-Matching Polytop Q^n_{2M} (= konvexe Hülle der Inzidenzvektoren von perfekten 2-Matching)

    perfektes 2-Matching = Kantenmenge, so dass jeder Knoten mit genau 2 Kanten dieser Menge inzident ist

- Für das perfekte 2-Matching Polytop ist eine lineare Beschreibung für beliebige Graphen bekannt (Edmonds 1965)

  $Q^n_{2M} = \{ x \in \mathbb{R}^E \mid 0 \leq x_e \leq 1$ für alle Kanten e

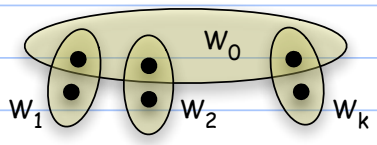
  $x(\delta(v)) = 2$ für alle Knoten v

  $\sum_{i=0}^{k} x(E(W_i)) \leq |W_0| + \frac{1}{2}(k-1)$ k ungerade (8.4)

  mit $W_0, W_1, \ldots, W_k \subseteq V$

  $|W_0 \cap W_i| = 1 = |W_i - W_0|$,

  $W_i \cap W_j = \emptyset$ für $i, j > 0$ }



Beim TSP Polytop lassen sich die 2-Matching Ungleichungen zu Kamm-Ungleichungen verallgemeinern

G vollständig, $W_0, W_1, \ldots, W_k \subseteq V$ mit

$|W_0 \cap W_i| \geq 1 \quad i = 1, \ldots, k$

$|W_i - W_0| \geq 1 \quad i = 1, \ldots, k$

$W_i \cap W_j = \emptyset \quad$ für $i, j > 0$

$k \geq 3$, ungerade

Dann muss gelten

$$\sum_{i=0}^{k} x(E(W_i)) \leq |W_0| + \sum_{i=0}^{k} (|W_i| - 1) - \frac{k+1}{2} \quad (8.5)$$

Die Kamm-Ungleichungen haben Chvátal Rang 1 bzgl. des 2-Matching Polytops.

8.4 Satz (Ungleichungen für das TSP Polytop, Chvátal 1973, Grötschel & Padberg 1979)

Für das TSP Polytop Q^n_{TSP} gilt ab $n = 6$

(a) $\dim(Q^n_{TSP}) = m - n$

(b) die Ungleichungen (8.3) definieren Facetten für jede Kante e

(c) die Subtour-Eliminationsbedingungen (8.3) bilden Facetten wenn $3 \leq |W| \leq n-3$

(d) alle Kammungleichungen (8.5) bilden Facetten

ohne Beweis ❑

8.5 Bemerkung

(a) - (d) liefert keine vollständige lineare Beschreibung, und es ist auch keine bekannt.

Für Q^{10}_{TSP} sind ca. 50 Milliarden verschiedene Facetten bekannt, für Q^{120}_{TSP} sind es mehr als 10^{179}

=> vollständige lineare Beschreibungen werden zu groß um LP Algorithmen direkt zu verwenden

Aber Verwendung bei Branch & Cut möglich, d.h. durch Separierung nur die bzgl. der Zielfunktion notwendige Menge von Schnittebenen generieren, die ausreicht, eine optimale Ecke vollständig zu beschreiben (siehe nächsten Abschnitt).

- Ziel dieses Abschnitts
  - Vorstellung eines konkreten Branch & Cut Algorithmus für TSP
  - Beispiele für polynomiale Separierung
  - Beispiele für den Nachweis, dass Ungleichungen Facetten definieren

- **Ein Branch & Cut Algorithmus für TSP**
  - Der Algorithmus startet mit dem LP

    (P) max $c^T x$ unter

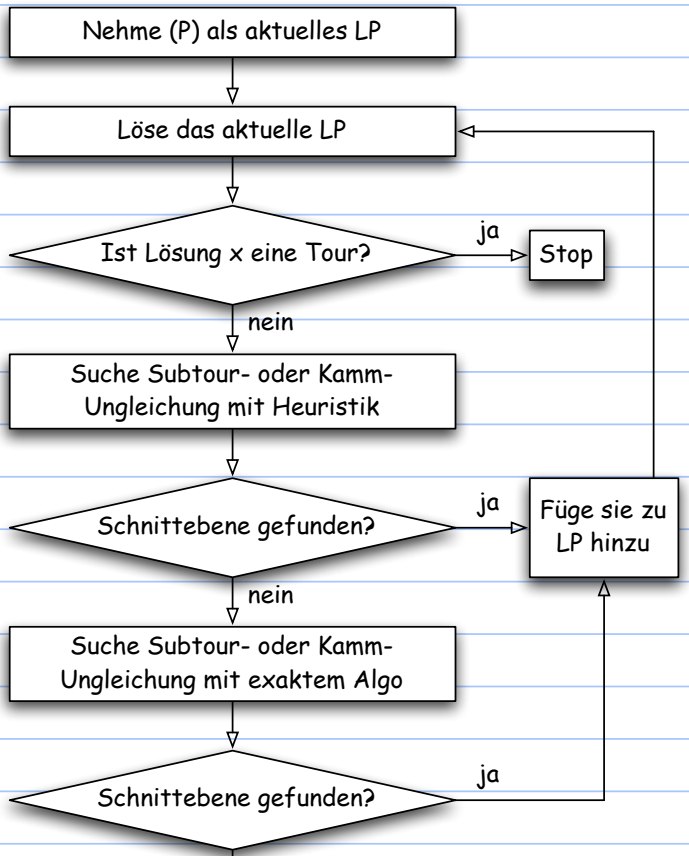
    $x(\delta(v)) = 2$ für alle Knoten v (8.1)

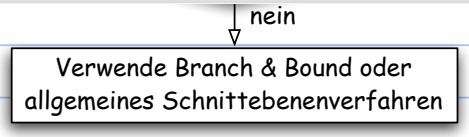
    $0 \le x_e \le 1$ für alle Kanten e (8.3)

    und fügt dann zunächst Subtour-Eliminationsbedingungen und danach Kammungleichungen ein (zunächst mit einer schnellen Heuristik, dann exakt).

    Wird keine dieser Ungleichungen mehr gefunden, wird Branch und Bound oder ein allgemeines Schnittebenenverfahren verwendet

  - Ein Flussdiagramm des Algorithmus





- In den Knoten des B&B Baumes wird dieses Prinzip dann wieder angewendet. Meist ist man dann in einigen zusätzlichen Iterationen fertig, da bereits gute untere Schranken bekannt sind.
- Exakt lösbare Größenordnung mit diesem Verfahren: mehrere Tausend Knoten.
  Mit Zusatztechniken kommt man bis zu 100.000 und mehr Knoten, vgl Concorde
  - http://www.tsp.gatech.edu/concorde/index.html

- Branch & Cut motiviert
  - Konstruktion schneller Separierungsalgorithmen
  - Suche nach gültigen Ungleichungen bzw. Facetten

- **Konstruktion schneller Separierungsalgorithmen**
  - dies führt oft wieder auf kombinatorische Optimierungsprobleme

- Beispiel 1: Ketten-Ungleichungen für das Antikettenpolytop
  - Sie haben die Form

    $x(K) \leq 1$ für jede Kette K

    Des Separierungsproblems bzgl. der Kettenungleichungen für einen Vektor x^* kann dann wie folgt gelöst werden:

    (1) berechne die längste Kette K^* bzgl. der Gewichtung x^*

    (2) ist $x^*(K^*) > 1$, so ist $x(K^*) \leq 1$ eine x^* separierende Ketten-Ungleichung
    ist $x^*(K^*) \leq 1$, so erfüllt x^* alle Ketten-Ungleichungen

    Die längste Kette kann im Kantendiagramm als längster Weg berechnet werden (ist polynomial, da das Kantendiagramm azyklisch ist)

- Beispiel 2: Subtour-Eliminationsbedingungen für das TSP Polytop
  - dabei setzen wir voraus, dass folgende Ungleichungen erfüllt sind

    $x(\delta(v)) = 2$ für alle Knoten v Gradungleichungen (8.2)

    $0 \leq x \leq 1$ Variablenbeschränkungen (8.3)

    erfüllt sind (was effizient überprüft werden kann)

8.6 Lemma (Separierung der Subtour-Eliminationsbedingungen)

x* erfülle die Ungleichungen (8.1) und (8.2)

Dann gibt es eine x* separierende Subtour-Eliminationsbedingung

<=> es gibt einen Schnitt $\delta(X)$ mit $x^*(\delta(X)) < 2$

Beweis

Sei X eine Knotenmenge, $\emptyset \neq X \neq V$. Dann ist

$x^*(\delta(X)) = \sum_{v \in X} \sum_{e \in \delta(v)} x_e^* - 2 \sum_{e \subseteq X} x_e^*$

$= \sum_{v \in X} 2 - 2 \sum_{e \subseteq X} x_e^*$ wegen (8.1)

$= 2|X| - 2 \sum_{e \subseteq X} x_e^* = 2|X| - 2x^*(E(X))$

Also $x^*(\delta(X)) + 2x^*(E(X)) = 2|X|$ (*)

"<=>"

x* verletze die Subtour-Eliminationsbedingung bzgl. X

<=> $x^*(E(X)) > |X| - 1$ <=> $2x^*(E(X)) > 2|X| - 2$ <=> $x^*(\delta(X)) < 2$ mit (*) ❑

8.7 Folgerung (Separierung der Subtour-Eliminationsbedingungen)

Die Separierung der Subtour-Eliminationsbedingungen bei Gültigkeit der Gradungleichungen (8.1) führt auf

die Berechnung eines Schnittes $\delta(X)$ mit minimaler Kapazität und kann durch eine Folge von s,t-Max-Fluss Problemen gelöst werden.

Ohne Beweis ❑

Suche nach gültigen Ungleichungen bzw. Facetten

Wir demonstrieren dies für das Antikettenpolytop

8.8 Satz (Facetten des Antikettenpolytops)

Für das Antikettenpolytop P_A gilt

$x_e = 0$ definiert eine Facette für jede Kante e

$x(K) = 1$ definiert eine Facette <=> K ist maximale Kette

Beweis

(1) $\dim(P_A) = |E|$, d.h, P_A ist volldimensional

Die Einheitsvektoren entsprechen den ein-elementigen Antiketten

wegen (1) ist eine Facette eine Stütz-Hyperebene H, deren Schnitt mit P_A die Dimension $|E| - 1$ hat

=> man muss $m := |E|$ Vektoren $x^0, \ldots, x^{m-1}$ in $H \cap P_A$ finden, so dass $x^i - x^0$ linear unabhängig sind (i

= 1, ..., m-1)

(2) $x_e = 0$ definiert eine Facette für jede Kante e

setze $x^0 := 0$, $x^i :=$ i-ter Einheitsvektor außer $i = e$

=> all diese Vektoren sind in der Hyperebene $x_e = 0$ enthalten

und die $x^i - x^0$ sind linear unabhängig

(3) $x(K) = 1$ definiert eine Facette <=> K ist maximale Kette

"<="

sei $K = \{ e_1, ..., e_k \}$ und $E-K = \{ e_{k+1}, ..., e_m \}$

K maximal => zu $e_j \in E-K$ gibt es $e_{i(j)} \in K$ mit $e_j, e_{i(j)}$ sind unvergleichbar

=> $\{e_1\}, ..., \{e_k\}, \{e_{k+1}, e_{i(k+1)}\}, ..., \{e_m, e_{i(m)}\}$ sind Antiketten

und ihre Inzidenzvektoren $y^0, ..., y^{m-1}$ liegen auf der Hyperebene $x(K) = 1$

$y^0, ..., y^{m-1}$ sind offenbar linear unabhängig

=> $y^i - y^0$ sind linear unabhängig => $x(K) = 1$ ist eine Facette

"=>"

Betrachte eine Kette K sodass $x(K) = 1$ eine Facette, aber K keine maximale Kette ist.

Sei dann K' eine maximale Kette mit $K \subset K'$, und seien $e \in K$ und $e' \in K'-K$.

Claim: $\{ x(K) = 1 \} \cap P_{\mathcal{A}} \subseteq \{ x(K') = 1 \} \cap P_{\mathcal{A}}$

sei $x \in P_{\mathcal{A}}$ und $x(K) = 1$

$K \subseteq K'$ => aus $x(K) = 1$ folgt $x(K') \geq 1$

aber wegen $x \in P_{\mathcal{A}}$ folgt $x(K') = 1$ => $x \in \{ x(K') = 1 \} \cap P_{\mathcal{A}}$

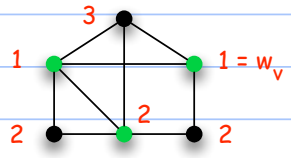
Facetten H definieren inklusions-maximale Mengen $H \cap P_{\mathcal{A}}$

Aber der Inzidenzvektor zu $\{e'\}$ liegt in $\{ x(K') = 1 \} \cap P_{\mathcal{A}} - \{ x(K) = 1 \} \cap P_{\mathcal{A}}$,

ein Widerspruch ❑

9.1 Einfaches Runden und Verwendung von dualen Lösungen

- Ziele dieses Kapitel
  - An 3 ausgewählten Techniken zeigen, dass LP-Theorie fortgeschrittene Methoden zur Konstruktion von Approximationsalgorithmen bereitstellt.
  - Bitte Approximationsalgorithmen aus ADM I unbedingt wiederholen!
- Ziele dieses Abschnittes
  - Approximationsalgorithmen basierend auf dem Lösen eines LPs mit anschließendem Runden
  - Beweis von Approximationsgüten durch Verwendung von LP-Dualität und dualen Lösungen
- Als Beispielproblem betrachten wir WEIGHTED VERTEX COVER (WVC)
  - Instanz
    - ein ungerichteter Graph G mit Knotengewichten $w_v \geq 0$
  - Aufgabe
    - Bestimme ein vertex cover C von G mit minimalem Gewicht $\sum_{v \in C} w_v$
  - Beispiel



C = grüne Knoten

Einfaches Runden (bei Minimierungsproblemen)

Algorithmus einfaches Runden

erstelle eine (geeignete) IP-Formulierung (IP) der Probleminstanz I

löse die LP-Relaxation (LP) von (IP) in polynomialer Zeit

Dies ist erforderlich, da Approximationsalgorithmen in polynomialer Zeit (in der Kodierungslänge <I> von I) laufen müssen

Wenn (LP) nur polynomial viele Variable und Ungleichungen in der Kodierungslänge von I hat, so kann es in polynomialer Zeit mit einem polynomialen LP-Algorithmus (vgl. Kapitel 10) gelöst werden.

Oft hat (LP) jedoch exponentiell viele Ungleichungen. Dann muss gezeigt werden, dass das Separierungsproblem für diese Ungleichungen polynomial lösbar ist. Hieraus folgt wegen Satz 7.26, dass (LP) in polynomialer Zeit gelöst werden kann.

runde die fraktionale Optimallösung von (LP) auf eine zulässige Lösung von (IP)

Dieses Runden ist problemabhängig. I.A. ist nicht klar ob es überhaupt geht

9.1 Lemma (Approximationsgüte beim einfachen Runden)

Sei A(I) die durch Runden erhaltene zulässige Lösung von (IP).

Sei OPT(I) eine optimale Lösung des Ausgangsproblem und LP(I) eine optimale Lösung der LP-Relaxation.

Gilt

$A(I) \leq \rho \cdot LP(I)$ für jede Instanz I,

so ist der Algorithmus "Einfaches Runden" ein ρ-Approximationsalgorithmus

Bemerkung: wie bei Approximationsalgorithmen üblich, verwenden wir A(I), OPT(I) usw. sowohl für die Lösung wie auch für den Wert der Lösung.

Beweis

Da (LP) eine Relaxierung von (IP) ist, gilt $LP(I) \leq IP(I) = OPT(I)$

$\Rightarrow A(I) \leq \rho \cdot LP(I) \leq \rho \cdot OPT(I)$ ❑

Anwendung auf WVC

IP-Formulierung (IP)

- Führe 0/1-Variable x_v ein mit $x_v = 1$:<=> $v \in C$
- Dann ist WVC äquivalent zu folgendem IP

  min $\sum_v w_v x_v$

  unter $x_u + x_v \geq 1$ für jede Kante $e = (u,v)$ von G

  $x_v \in \{0, 1\}$ für jeden Knoten v von G

- Die LP-Relaxation (LP) von (IP)

  min $\sum_v w_v x_v$

  unter $x_u + x_v \geq 1$ für jede Kante $e = (u,v)$ von G

  $x_v \geq 0$ für jeden Knoten v von G

- $x_v \geq 0$ reicht, da sich in der optimalen LP Lösung $x_v \leq 1$ automatisch wegen $w_v \geq 0$ ergibt.
- (LP) hat nur polynomial viele Ungleichungen und Variable und kann in polynomialer Zeit mit einem polynomialen LP-Algorithmus (vgl. Kapitel 10) gelöst werden
- Das Runden
  - sei x' eine Optimallösung von (LP)
  - runde x' zu x^* wie folgt:

$$x_v^* := \begin{cases} 1 & \text{if } x_v' \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

- Dann gilt:
- x^* ist zulässig für (IP), d.h. ein vertex cover
  - sei (u,v) eine Kante von G
  - => $x_u' + x_v' \geq 1$ => $x_u' \geq 0.5$ oder $x_v' \geq 0.5$ => $x_u^* = 1$ oder $x_v^* = 1$
  - => die Kante (u,v) wird von x^* überdeckt
- $A(I) \leq 2 \cdot LP(I)$
  - $x_v^* \leq 2x_v'$ => $A(I) \leq 2 \cdot LP(I)$ da $w_v \geq 0$
- Damit ist der Algorithmus "Einfaches Runden" ein 2-Approximationsalgorithmus für WVC

Die Verwendung von dualen Lösungen (bei Minimierungsproblemen)

9.2 Lemma (Verwendung von dualen Lösungen in Approximationsalgorithmen)

Sei (D) das duale LP zur LP-Relaxation (LP) von (IP).

Sei dual(I) eine zulässige Lösung von (D) zur Instanz I.

Sei A ein polynomialer Algorithmus, der eine zulässige Lösung A(I) von (IP) konstruiert mit

$A(I) \le \rho \cdot dual(I)$ für jede Instanz I

Dann ist A ein ρ-Approximationsalgorithmus

Beweis:

schwacher Dualitätssatz => $dual(I) \le LP(I)$

(LP) ist Relaxation von (IP) => $LP(I) \le OPT(I)$

Also ist $A(I) \le \rho \cdot dual(I) \le \rho \cdot OPT(I)$ ❑

Bemerkung

Im Gegensatz zum einfachen Runden muss man bei der Verwendung von dualen Lösungen kein LP lösen. Es reicht, das der Algorithmus eine zulässige Lösung $A(I)$ von (IP) konstruiert. Die duale Lösung $dual(I)$ braucht man nur im Beweis der Ungleichung

$A(I) \le \rho \cdot dual(I)$

und nicht im Algorithmus.

Anwendung auf WVC (Bar-Yehuda & Even 1981)

Die LP-Relaxation (LP) von WVC (vgl. oben) ist

$$\begin{aligned} \min \quad & \textstyle\sum_v w_v x_v \\ \text{unter} \quad & x_u + x_v \ge 1 \qquad \text{für jede Kante } e = (u,v) \text{ von } G \\ & x_v \ge 0 \qquad \text{für alle Knoten } v \text{ von } G \end{aligned}$$

Das zugehörige duale LP (D) hat Variable y_e für jede Kante e von G und lautet

$$\begin{aligned} \max \quad & \textstyle\sum_{e \in E} y_e \\ \text{unter} \quad & \textstyle\sum_{e \in \delta(v)} y_e \le w_v \qquad \text{für jeden Knoten } v \text{ von } G \\ & y_e \ge 0 \qquad \text{für alle Kanten } e \text{ von } G \end{aligned}$$

Man sucht also Kantengewichte $y_e \ge 0$ so dass das Gesamtgewicht im Graphen so groß wie möglich wird, aber in jedem "Stern" $\delta(v)$ das Gewicht höchstens w_v ist (ein w-packing mit maximalem Wert)

Der Spezialfall $w_v = 1$ für alle Knoten

Algorithmus G (Gavril, 1974, vgl. ADM I)

Input

eine Instanz I von VERTEX COVER

Output

ein vertex cover $G(I)$ mit $G(I) \le 2 \cdot OPT(I)$

Methode

- berechne ein $\subseteq$-maximales Matching M in G
- setze U := von jeder Matching Kante aus M beide Endpunkte
- return U

- Beweis der Approximationsgüte durch Verwendung von dualen Lösungen
  - Die im Algorithmus berechnete Menge U ist ein vertex cover
    - ansonsten wäre M kein $\subseteq$-maximales Matching
  - M ist eine dual zulässige Lösung, M = dual(I)
    - klar, da in jedem Stern $\delta(v)$ höchstens eine Matching Kante liegt
  - $G(I) = |U| = 2|M| = 2 \cdot \text{dual}(I) \Rightarrow G(I) \leq 2 \cdot \text{dual}(I) \Rightarrow$ Güte 2 mit Lemma 9.2 ❑

- Der allgemeine Fall beliebiger Gewichte $w_v \geq 0$
  - Nenne einen Knoten v gesättigt, wenn $\sum_{e \in \delta(v)} y_e = w_v$

- Algorithmus PACK
  - Input

    - eine Instanz I von WVC mit $E \neq \emptyset$ und o.B.d.A. $w_v > 0$ für alle v

      // Knoten v mit $w_v = 0$ wird man direkt nehmen und PACK nur auf den Restgraphen anwenden
  - Output
    - ein vertex cover PACK(I) mit $\text{PACK}(I) \leq 2 \cdot \text{OPT}(I)$
  - Methode
    - setze $C := \emptyset$ und $y_e = 0$ für alle Kanten e
    - repeat
      - wähle eine Kante e
      - erhöhe den Wert der Dualvariablen y_e bis einer (oder beide) Endpunkte von e gesättigt ist
      - füge die gesättigten Endpunkte von e zu C hinzu
      - lösche den/die gesättigten Endpunkte von e und alle inzidenten Kanten
    - until keine Kanten übrig
    - return C

- Beweis der Approximationsgüte durch Verwendung von dualen Lösungen
  - Die im Algorithmus berechnete Menge C ist ein vertex cover

- eine Kante e wird nur gelöscht wenn ein Endpunkt u gesättigt ist
  => $u \in C$ und u überdeckt die Kante e
- Die Kantengewichte am Ende des Algorithmus bilden eine dual zulässige Lösung dual(I)
  - klar, da im gesamten Algorithmus $y_e \geq 0$ und $\sum_{e \in \delta(v)} y_e \leq w_v$ gilt
  - $PACK(I) = \sum_{v \in C} w_v \leq 2 \cdot \sum_{e \in E} y_e = 2 \cdot dual(I)$
  - => Güte 2 mit Lemma 9.2
  - interpretiere die Erhöhung von y_e um k als Bezahlung von k \$ an jeden Endpunkt von e
    => einem Knoten v wurde insgesamt w_v gezahlt wenn er in die Menge C kommt
    => $\sum_{v \in C} w_v \leq$ Gesamtzahlung an alle Knoten $= 2 \sum_{e \in E} y_e$ ❑

- Ziele dieses Abschnittes
  - Approximationsalgorithmen basierend auf dem Lösen eines LPs mit anschließendem randomisierten Runden, d.h., gerundet wird mit Wahrscheinlichkeiten, die sich aus der Optimallösung des LP ergeben
  - Als Illustration dient MAX SAT, vgl. ADM I, Kapitel 9.4

- Eine triviale Randomisierung für MAX SAT (vgl. ADM I, Kapitel 9.4)
  - Algorithmus Randomize (Johnson 1974)
    - Input
      - eine Instanz von MAX SAT
      - pro Klausel Z mindestens k Literale ($k \geq 1$)
      - Gewichte $c(Z)$ pro Klausel Z
    - Output
      - eine zufällige Belegung mit erwarteter Güte
        $E[\sum_Z c(Z)] \geq (1 - 1/2^k) \cdot OPT(I)$
    - Methode
      - werfe für jede Boolesche Variable x_i eine Münze und setze

- x_j := TRUE falls Kopf
- x_j := FALSE falls Zahl
- return die entstehende zufällige Belegung

Dieser Algorithmus ist gut für $k \geq 2$ (liefert in Erwartung mindestens 3/4 optimalen Gewichtes), aber schlecht für $k = 1$ (nur 1/2 des optimalen Gewichtes).

Randomisierung auf Basis einer LP Relaxation

Generelles Vorgehen beom Randomisierten Runden (Raghavan & Thompson 1987)

1. Modelliere das Problem als IP — Variable $x_j \in \{0, 1\}$
2. Relaxiere das IP zu einem LP — $0 \leq x_j \leq 1$
3. Löse das LP optimal — Werte x_j'
4. Runde randomisiert

   setze $x_j = 1$ mit Wahrscheinlichkeit x_j'
5. Zeige dass der so erhaltene Vektor x

   zulässig für das IP ist

eine gute erwartete Güte hat

Konkretes Vorgehen bei MAX SAT (Goemans & Williamson 1993)

Das IP

0/1 Variable y_i = Wahrheitswert der Booleschen Variablen x_i

0/1 Variable z_j = Wahrheitswert der Klausel C_j

T_j = Menge der unnegierten Variablen in Klausel C_j

F_j = Menge der negierten Variablen in Klausel C_j

Das IP lautet dann

$$\max \sum_j w_j z_j$$

$$\text{unter} \sum_{i \in T_j} y_i + \sum_{i \in F_j} (1 - y_i) \geq z_j$$

$$y_i \in \{0, 1\}, \quad z_j \in \{0, 1\}$$

Das randomisierte Runden

Sei (y^*, z^*) die Optimallösung der LP Relaxation des IP

Nutze y^* für randomisiertes Runden, d.h. setze

x_i := TRUE mit Wahrscheinlichkeit y_i^*

x_i := FALSE mit Wahrscheinlichkeit $1-y_i^*$

Dies liefert trivialerweise eine Belegung der Instanz mit Wahrheitswerten für die Booleschen Variablen.

Natürlich müssen nicht alle Klauseln erfüllt sein.

Die Gütegarantie

Betrachte o.B.d.A. die Klausel $C_j = x_1 \vee x_2 \vee \ldots \vee x_k$ (analog für negierte Variable und andere Indizes).

Dann gilt

$$\text{W'keit}[C_j \text{ erfüllt}] = 1 - \prod_{i=1}^{k}(1-y_i^*)$$

$$\geq 1 - \left(1 - \frac{1}{k}\sum_{i=1}^{k} y_i^*\right)^k \quad \text{wegen geometrisches Mittel} \leq \text{arithmetisches Mittel}$$

$$\geq 1 - \left(1 - \frac{1}{k} z_j^*\right)^k \quad \text{wegen LP Ungleichung}$$

Nun ist

$$f(z) := 1 - \left(1 - \frac{1}{k} z\right)^k \quad \text{konkav in } z$$

und

$$g(z) := \left(1 - \left(1 - \frac{1}{k}\right)^k\right) z \quad \text{linear in } z$$

Also gilt $f(z) \geq g(z)$ auf dem ganzen Intervall $[0,1]$, wenn $f(z) \geq g(z)$ für die Intervall-Endpunkte $z = 0$ und $z = 1$.

Überprüfung $z = 0$: $f(0) = 0$, $g(0) = 0$

Überprüfung $z = 1$: $f(1) = g(1)$

Also gilt

$$\text{W'keit}[C_j \text{ erfüllt}] \geq \left(1 - \left(1 - \frac{1}{k}\right)^k\right) z_j^*$$

$\Rightarrow$ E(Gesamtgewicht aller erfüllten Klauseln)

$$\geq \min_k \left(1 - \left(1 - \frac{1}{k}\right)^k\right) \sum_j w_j z_j^* \geq \left(1 - \frac{1}{e}\right) \sum_j w_j z_j^*$$

$\geq \left(1 - \frac{1}{e}\right)$ OPT(I) da z^* Optimallösung der LP-Relaxation ist

$\approx 0{,}623 \cdot OPT(I)$

- Dieser Algorithmus erreicht also auch für Instanzen mit nur einem Literal in einer Klausel in Erwartung mindestens 0,623 des Optimalgewichtes.

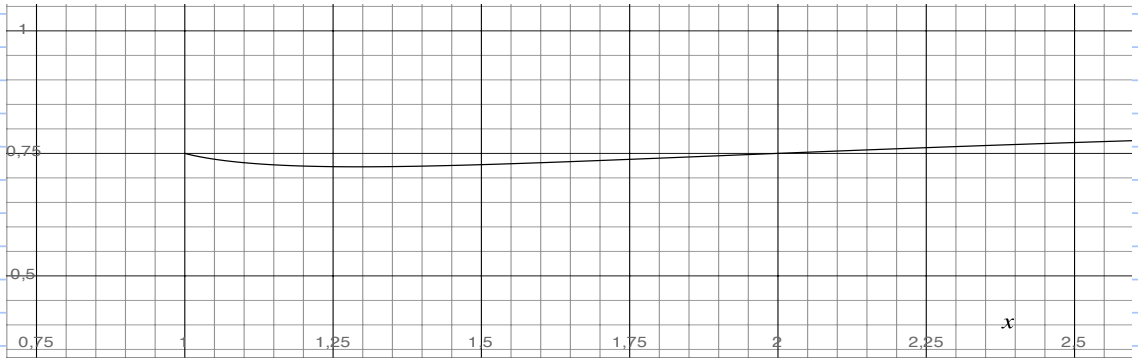
- Kombination der beiden Algorithmen für MAX SAT
  - Werfe eine Münze um zu entscheiden, welcher der beiden Algorithmen (Johnson oder Randomisiertes Runden) genommen wird und führe den gewählten ALgorithmus aus.
  - Dies ist wieder ein randomisierter Algorithmus mit einer erwarteten Güte von (3/4)·OPT(I)
    - Beweis:
      - Betrachte eine Klausel C_j mit k Literalen
        Dann ist

$$\text{W'keit}[C_j \text{ erfüllt}] = \frac{1}{2}\left(1 - \frac{1}{2^k}\right) + \frac{1}{2}\left(1 - \left(1 - \frac{1}{k}\right)^k\right) z_j^*$$

$$\geq \frac{1}{2}\left(1 - \frac{1}{2^k}\right) z_j^* + \frac{1}{2}\left(1 - \left(1 - \frac{1}{k}\right)^k\right) z_j^* \quad \text{da } 0 \leq z_j^* \leq 1$$

$$= f(k) \cdot z_j^* \text{ mit } f(k) := \frac{1}{2}\left(1 - \frac{1}{2^k}\right) + \frac{1}{2}\left(1 - \left(1 - \frac{1}{k}\right)^k\right)$$

- Nun ist $f(1) = 3/4$ und $f(x) \geq 3/4$ auf dem Intervall $[2,\infty]$ (Kurvendiskussion)



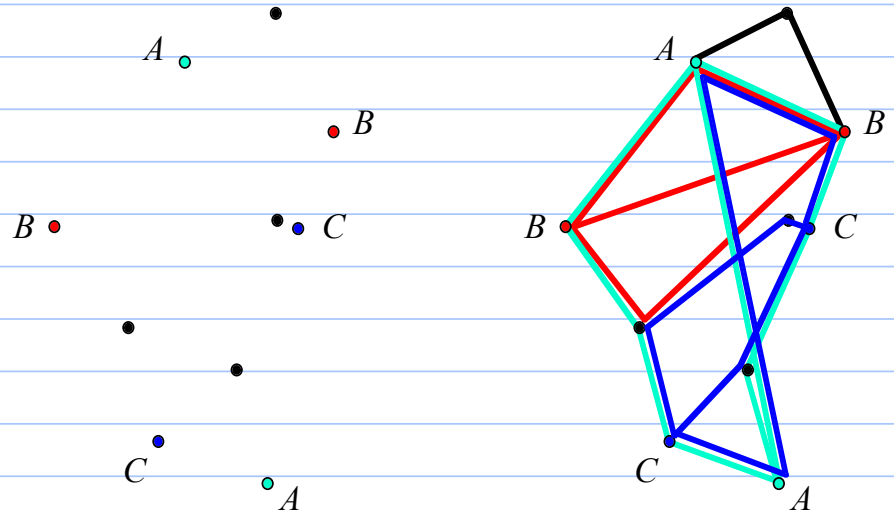
=> E(Gesamtgewicht aller erfüllten Klauseln) $\geq 3/4 \sum_j w_j z_j^* \geq 3/4$ OPT(I) ❑

- Ziele dieses Abschnittes
  - Einführung des primal-dualen Schemas zur Konstruktion von Approximationsalgorithmen
  - Als Beispiel dient das Problem des Netzwerk Design

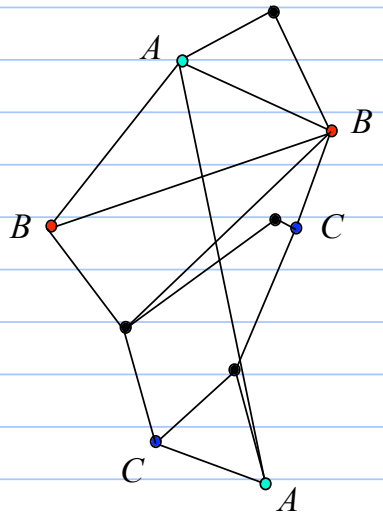
- **Das Netzwerk Design Problem**
  - Instanz
    - ein ungerichteter Graph $G = (V, E)$
    - Kantenkosten $c_e \geq 0$
    - Verbindungs-Anforderungen r_{ij} für je 2 Knoten i, j
  - Gesucht
    - eine Kantenmenge $F \subseteq E$ mit minimalen Kosten $\sum_{e \in F} c_e$ so dass
    - $G' = (V, F)$ enthält für je 2 Knoten i, j mindestens r_{ij} paarweise kantendisjunkte Wege zwischen i und j

  - Das Netzwerk Design Problem ist NP-schwer (Karp 1972).
    Es tritt auf bei der Konstruktion preiswerter Netzwerke, die gegen den Ausfall einzelner Kanten abgesichert sind.

- Konstruktion eines VPN als Spezialfall (vgl. Abschnitt 4.4)
  - Beispiel
    - Gegebene Knoten mit Anforderungen r_{ij} = 3 zwischen farbigen Knoten
      r_{ij} = 2 sonst
    - Eine Lösung des Netzwerk Design Problem



- Die Lösung "überlebt" den Ausfall von bis zu 2 Kanten auf Verbindungen zwischen farbigen Knoten und von einer Kante von Verbindungen mit einem schwarzen Endpunkt. Sie war das Beispiel VPN für das Disjunkte Wege Problem in Abschnitt 4.4.



- Eine Formulierung des Netzwerk Design Problem als IP
  - setze $f(S) := \max \{ r_{ij} \mid i \in S, j \notin S \}$ für jede Knotenmenge $S \neq \emptyset, V$ (Bedarf von S)
  - führe 0/1 Variable x_e für die Wahl der Kante e ein
  - (IP):

    $\min \sum_e c_e x_e$

$\sum_{e \in \delta(S)} x_e \geq f(S)$ für alle $\emptyset \neq S \subset V$ (Schnittbedingung)

$x_e \in \{ 0, 1 \}$ für alle Kanten e

- x ist Lösung von (IP) <=> $F = \{ e \in E \mid x_e = 1 \}$ ist Lösung des Netzwerk Design Problem
  - "<="
    - trivial
  - "=>"
    - Max Fluss Min Schnitt Satz (angewendet auf ein Paar i, j mit Kantenkapazitäten 1) + Schnittbedingung
      => es gibt einen Fluss der Stärke r_{ij} von i nach j
      => es gibt einen ganzzahligen Fluss der Stärke r_{ij} von i nach j
      => es gibt r_{ij} paarweise kantendisjunkte Wege von i nach j
      => Die Bedingungen im (IP) sind auch hinreichend ❑

- Einige Spezialfälle
  - Kürzeste s,t-Wege
    - $r_{st} = 1$, $r_{ij} = 0$ sonst
      $f(S) = 1$ falls $|S \cap \{ s, t \}| = 1$, $f(S) = 0$ sonst

- Minimum spanning tree
  - $r_{ij} = 1$ für alle Paare i, j

    $f(S) = 1$ für alle $\emptyset \neq S \subset V$, $f(S) = 0$ sonst
- Minimum Steiner tree
  - $r_{ij} = 1$ für alle $i, j \in T$ (T = Menge der zu verbindenden Terminale)

    $f(S) = 1$ falls $S \cap T \neq \emptyset$ und $T - S \neq \emptyset$, $f(S) = 0$ sonst
- Verallgemeinertes Steiner Baum Problem
  - :<=> $f(S) \in \{ 0, 1 \}$

Das primal-duale Schema

- Nutzt die Bedingungen vom komplementären Schlupf wie der primal-duale Algorithmus zur Lösung von LPs in Kapitel 6.

  Zur Erinnerung (in der hier benötigten Form):
  - Ausgangspunkt sind ein LP

    (P) $\min \sum_j c_j x_j$ unter $\sum_j a_{ij} x_j \geq b_i$ für alle i, $x_j \geq 0$ für alle j

    und das dazugehörige duale

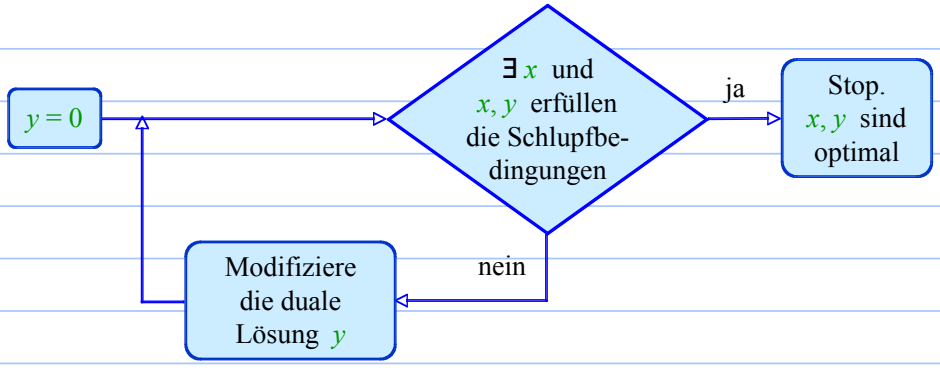
(D) $\max \sum_i b_i y_i$ unter $\sum_i a_{ij} y_i \leq c_j$ für alle j, $y_i \geq 0$ für alle i

- Die Bedingungen vom komplementären Schlupf lauten dann

  $x_j > 0 \Rightarrow \sum_i a_{ij} y_i = c_j$ (primale Schlupfbedingung)

  $y_i > 0 \Rightarrow \sum_j a_{ij} x_j = b_i$ (duale Schlupfbedingung)
- Ein primal zulässiges x und ein dual zulässiges y sind optimal

  <=> x und y erfüllen diese Bedingungen (Satz 4.4)
- Der primal duale Algorithmus in Kapitel 6 durchläuft dann folgende Schleife



- Das primal-duale Schema für Approximationsalgorithmen
  - Modelliere das Problem als IP
  - Relaxiere das IP zu einem LP
  - Relaxiere die duale Schlupfbedingung $(y_i > 0 \Rightarrow \sum_j a_{ij}x_j = b_i)$
  - Konstruiere über die Schleife eine zulässige Lösung x für das IP und eine dual zulässige Lösung y
  - Zeige dass x und y die Ungleichung $\sum_j c_j x_j \leq \alpha \cdot \sum_i b_i y_i$ erfüllen

    $\Rightarrow$ α-Approximation wegen Lemma 9.2

- Das primal-duale Paar für das Netzwerk Design Problem mit $f(S) \in \{0, 1\}$
  - IP:
    - min $\sum_e c_e x_e$

      $\sum_{e \in \delta(S)} x_e \geq f(S)$ für alle $\emptyset \neq S \subset V$ (Schnittbedingung)

      $x_e \in \{0, 1\}$ für alle Kanten e
  - LP Relaxation = primales LP
    - min $\sum_e c_e x_e$

      $\sum_{e \in \delta(S)} x_e \geq f(S)$ für alle $S \subset V$ mit $f(S) = 1$

      $x_e \geq 0$ für alle Kanten e

      Die Bedingung $x_e \leq 1$ kann fallen gelassen werden, da sie sich wegen $c_e \geq 0$ im Optimum von selbst ergibt
  - Duales LP
    - max $\sum_{S:f(S)=1} y_S$

      $\sum_{S:e \in \delta(S)} y_S \leq c_e$ für alle Kanten e

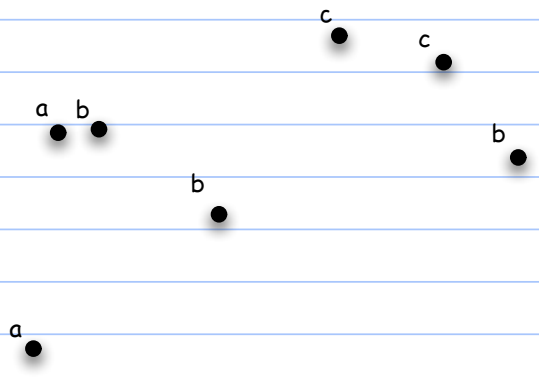
      $y_S \geq 0$ für alle Variablen y_S
  - Nenne eine Kante e gesättigt, wenn $\sum_{S:e \in \delta(S)} y_S = c_e$

    Die primale Schlupfbedingung sagt dann: $x_e > 0 \Rightarrow e$ gesättigt

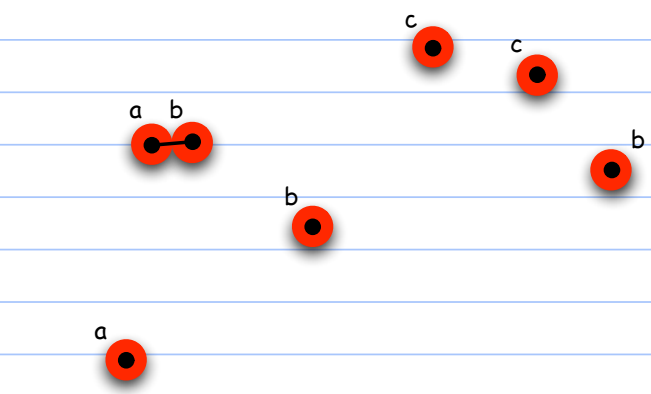
- Der primal duale Algorithmus für $f(S) \in \{0, 1\}$
  - Input
    - Instanz des Netzwerk Design Problem mit $f(S) \in \{0, 1\}$
  - Output
    - zulässige Lösung (V, A) des Netzwerk Design Problem mit Gütegarantie 2
  - Methode
    - Initialisiere alle Dualvariablen $y_S := 0$

- Initialisiere die primale Lösung (als Kantenmenge A) $A := \emptyset$
- while A ist keine zulässige Lösung do
  - sei C die Menge aller Zusammenhangskomponenten S im Graph (V, A) der bereits gewählten Kanten A mit $f(S) = 1$
  - erhöhe y_S für alle $S \in C$ um denselben Betrag bis eine Kante $e \notin A$ gesättigt ist
  - füge alle gesättigten Kanten zu A hinzu
- entferne unnötige Kanten aus A (mache A dadurch $\subseteq$-minimal zulässig) // cleanup step
- return A

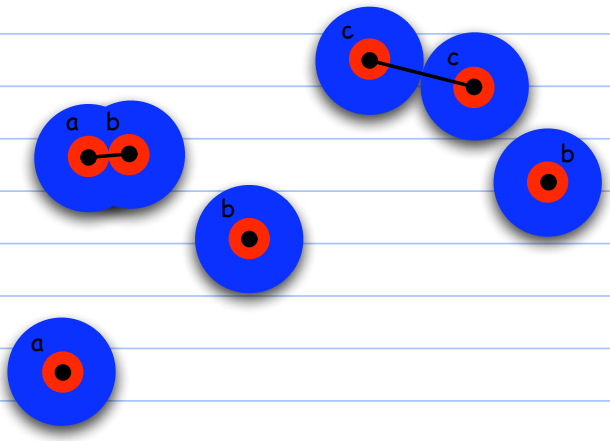
- Beispiel für Euklidische Abstände

- Dualvariable werde als Gräben um die Zusammenhangskomponenten $S \in C$ dargestellt
- Initialisierung: $A := \emptyset$, $y_S := 0$ für alle S
  - 

  - $r_{ij} = 1$ für i,j = a, i,j = b und i,j = c
  - => anfangs sind alle Zusammenhangskomponenten $S \in C$ einelementig
- Iteration 1



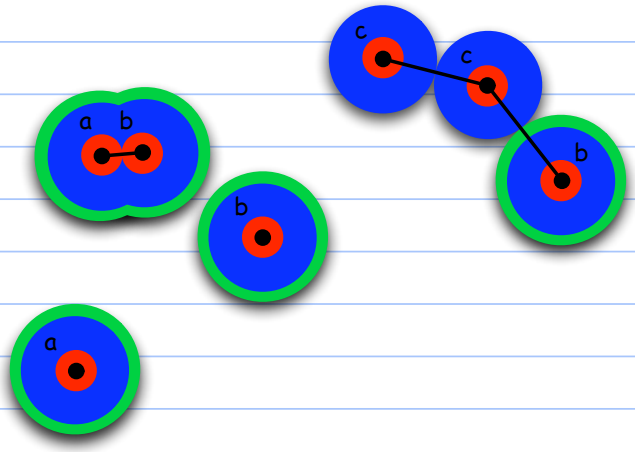
Iteration 2



Die jetzt entstandene Zusammenhangskomponente S (die beiden c-Knoten) hat f(S) = 0,

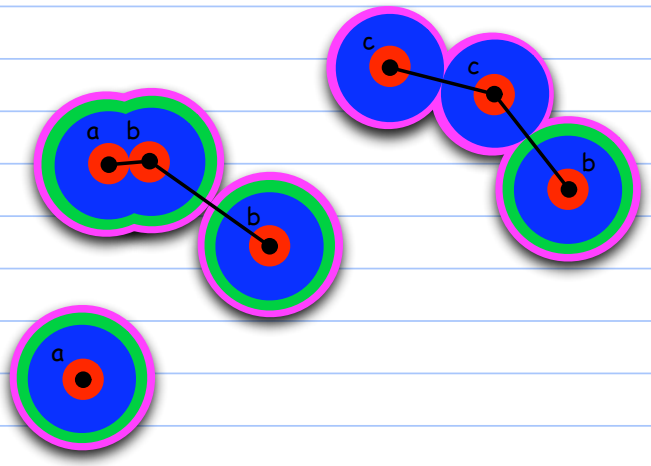
die zugehörige Dualvariable y_S wird also in der nächsten Iteration nicht wachsen.

Iteration 3

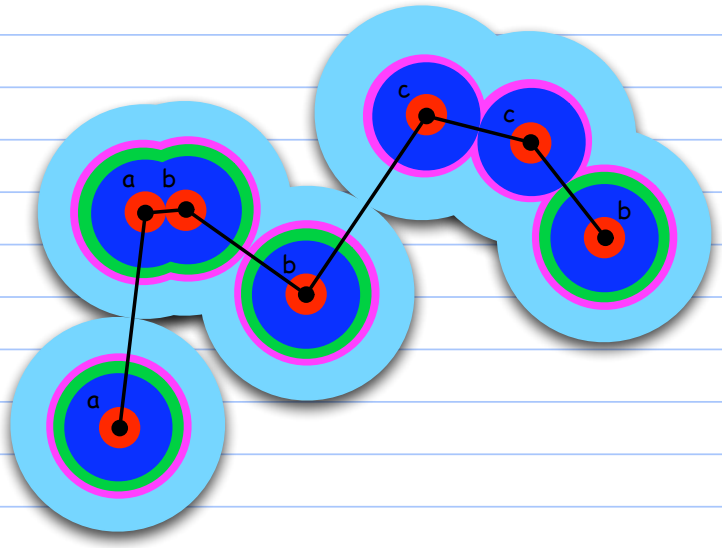


Die jetzt entstandene Zusammenhangskomponente S (die beiden c-Knoten und der b-Knoten) hat f(S) = 1, die zugehörige Dualvariable y_S wird also in der nächsten Iteration wachsen.

Iteration 4



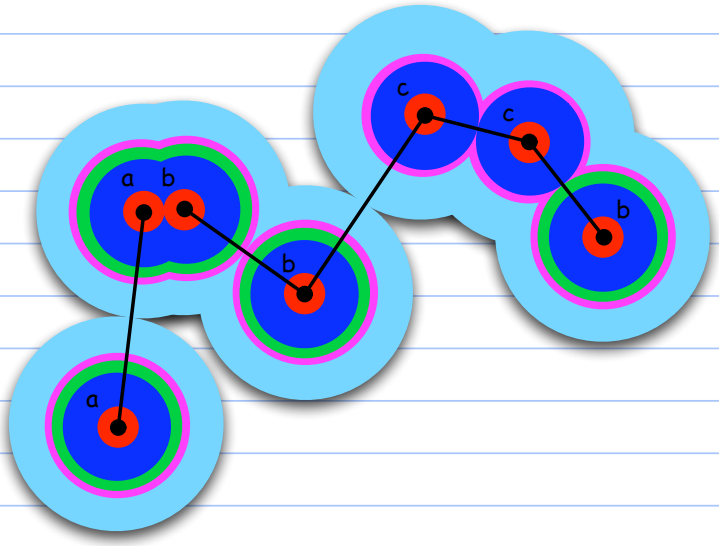
Iteration 5



In dieser letzten Iteration wurden gleichzeitig 2 Kanten gesättigt und zu A hinzugefügt.

Cleanup step

Die Kante zwischen a und b ist überflüssig und wird entfernt.



Gütegarantien im primal dualen Schema für f(S) ∈ { 0, 1 }

9.3 Satz (Güte und Laufzeit des primal dualen Algorithmus, Agarwal, Klein & Ravi 1991, Goemans & Williamson 1992)

Der primal duale Algorithmus für $f(S) \in \{0, 1\}$ kann mit einer Laufzeit von $O(n^2 \log n)$ implementiert werden.

Für die berechnete primale Lösung A gilt $\sum_{e \in A} c_e \leq 2 \cdot OPT(I)$, d.h. der Algorithmus hat eine Gütegarantie von 2.

Beweis

durch Lemma 9.4 und 9.5 (nur die Gütegarantie, nicht die Laufzeit) ❑

Die wesentliche kombinatorische Ungleichung

Der Algorithmus durchläuft die Iterationen $k = 1, 2, \ldots, K$

In Iteration k sei

A_k die Kantenmenge bei Eintritt in die Iteration

C_k die Menge der Zusammenhangskomponenten

ε_k der Wert um den alle y_S mit $S \in C_k$ vergrößert werden

9.4 Lemma (kombinatorische Ungleichung ergibt Approximationsgüte)

Gilt in jeder Iteration k des primal dualen Algorithmus

$\sum_{S \in C_k} |\delta(S) \cap D| \leq \alpha \cdot |C_k|$

für jede $\subseteq$-minimale zulässige Lösung D, die A_k enthält,

so ist er ein α-Approximationsalgorithmus.

Interpretation der kombinatorischen Ungleichung:

Jede $\subseteq$-minimale zulässige Obermenge D von A_k muss zu allen Schnitten $\delta(S)$ mit $S \in C_k$ Kanten hinzufügen.

Die Ungleichung sagt, dass die Anzahl dieser Kanten durch α mal Anzahl dieser Mengen S abgeschätzt werden kann.

Pro Menge S in Iteration k werden "im Mittel" also nur α Kanten benötigt.

Beweis für $f(S) \in \{0, 1\}$:

Sei y die vom Algorithmus konstruierte duale Lösung.

Sei A die vom Algorithmus konstruierte primale Lösung.

cleanup step => A ist enthalten in den Mengen D, für die die kombinatorische Ungleichung nach Vorraussetzung gilt.

=> kombinatorische Ungleichung gilt nach Voraussetzung auch für A

Für den Wert von A gilt:

$$\sum_{e\in A} c_e = \sum_{e\in A} \sum_{S: e\in\delta(S)} y_S \quad \text{wegen primaler Schlupfbedingung}$$

$$= \sum_{S:\delta(S)\cap A \neq \emptyset} |A\cap\delta(S)|\cdot y_S \quad \text{Umordnung der Summe}$$

$$= \sum_{S:\delta(S)\cap A \neq \emptyset} |A\cap\delta(S)|\cdot \sum_{k: S\in C_k} \varepsilon_k \quad \text{Wachstum von } y_S$$

$$= \sum_k \left(\sum_{S\in C_k} |A\cap\delta(S)| \right) \varepsilon_k \quad \text{Umordnung der Summe}$$

$$\leq \sum_k (\alpha\cdot |C_k|)\,\varepsilon_k \quad \text{wegen der kombinatorischen Ungleichung}$$

$$= \alpha \sum_k |C_k|\varepsilon_k = \alpha \sum_S y_S$$

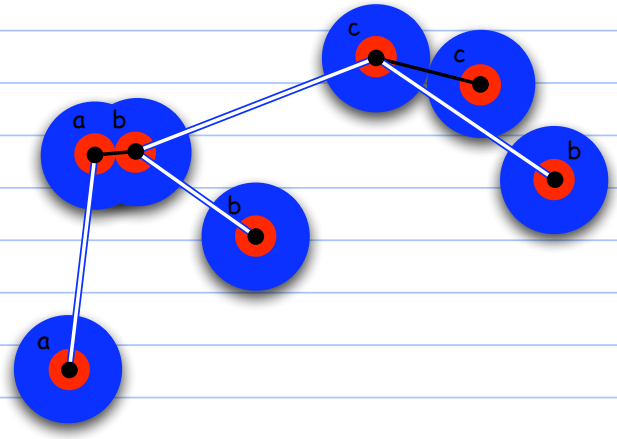
$\leq \alpha\cdot OPT(I)$ wegen Lemma 9.2 ❑

9.5 Lemma (Kombinatorische Ungleichung für $f(S) \in \{0, 1\}$)

Gilt $f(S) \in \{0, 1\}$, so erfüllt der primal duale Algorithmus die kombinatorische Ungleichung mit $\alpha = 2$.

Beweis:

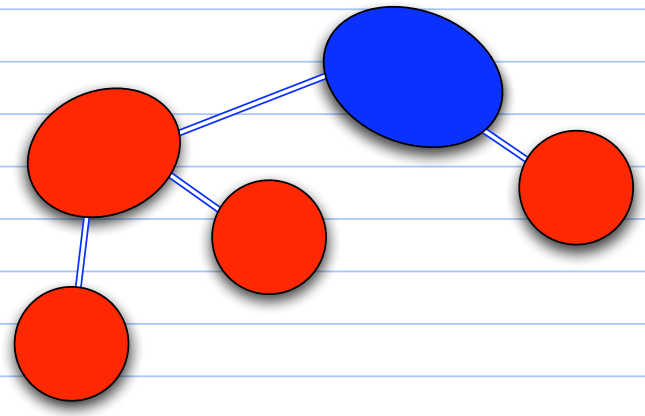
Betrachte Iteration k und eine $\subseteq$-minimale zulässige Obermenge D von A_k



Kontrahiere die Zusammenhangskomponenten in (V, A_k) zu Superknoten

D $\subseteq$-minimal zulässig, $f(S) \in \{0, 1\}$ => D bildet einen Wald im kontrahierten Graphen

Färbe die Superknoten, die zu Mengen $S \in C_k$ gehören rot, die anderen blau



Knoten $v = S \in C_k$ => $|\delta(S) \cap D|$ = Grad d(v)

=> die kombinatorische Ungleichung reduziert sich auf

$\sum_{v\ rot} d(v) \leq 2 \cdot (\#\ roter\ Knoten)$ im kontrahierten Graphen

Claim: kein blauer Knoten hat Grad 1

sonst wäre die entsprechende Kante notwendig für die Zulässigkeit

=> der Knoten müßte rot sein

Lasse blaue Knoten von Grad 0 weg (ändern nichts an der zu beweisenden Ungleichung).

Für den entstehenden Teilgraphen (immer noch ein Wald) gilt:

$\sum_{v\ rot} d(v) = \sum_{v\ rot\ oder\ blau} d(v) - \sum_{v\ blau} d(v)$

$\leq 2 \cdot (\#\ rot + \#\ blau) - \sum_{v\ blau} d(v)$ da der Graph ein Wald ist

$\leq 2 \cdot (\#\ rot + \#\ blau) - 2 \cdot (\#\ blau)$ wegen Claim

$= 2 \cdot (\#\ rot)$ ❑

Gütegarantieen im primal dualen Schema für beliebige Werte f(S)

Goemans, Mihail, Vazirani & Williamson 1993

Wende wiederholt eine Variation des primal dualen Algorithmus an

Man erhält einen $2 \cdot H(R)$-Approximationsalgorithmus mit

$R := \max_{ij} r_{ij}$ und

$H(R) := 1 + 1/2 + 1/3 + \ldots + 1/R \sim \log R$

Verhalten des primal dualen Algorithmus in der Praxis

- Steiner Bäume (Hall 1995)
  - 60 Instanzen von Beasley
    500 - 1000 Knoten, 600 - 60000 Kanten
    Im Mittel nur 7% entfernt vom Optimum
    Besser als Heuristiken auf großen Instanzen

- Verallgemeinerte Steiner Baum Probleme (Hu & Wein 1995)
  - 1000 zufällig erzeugte Instanzen, 32 - 64 Knoten
    Generell nur 5% vom Optimum entfernt

- Netzwerk Design (Mihail, Mostrel, Dean & Shallcross 1996)
  - Methode genutzt in Software Paket bei Bellcore
    (ITP/INPLANS CCS Network Topology Analyzer)
    Funktioniert angeblich gut in der Praxis, aber keine genaue Analyse verfügbar

- **Der Algorithmus von Jain für Netzwerk Design**

- Basiert auf einfachem Runden, liefert 2-Approximation für den allgemeinen Fall

- 9.6 Satz (Eigenschaften von Basislösungen der LP-Relaxation des Netzwerk Design Problem, Jain 1998)
  - Jede zulässige Basislösung x der LP-Relaxation des allgemeinen Netzwerk Design Problem hat eine Komponente e mit $x_e \geq 1/2$
  - Ohne Beweis ❑

- Hieraus resultiert folgender Algorithmus von Jain
  - Input
    - Instanz des allgemeinen Netzwerk Design Problem
  - Output
    - zulässige Lösung des Netzwerk Design Problem mit Gütegarantie 2
  - Methode
    - sei Q die LP-Relaxation der IP Formulierung der gegebenen Instanz
    - repeat forever
      - berechne eine optimale Basislösung x von Q

- if alle x_e sind ganzzahlig then return x
- runde x_e auf 1 für alle Kanten e mit $x_e \geq 1/2$
- modifiziere Q durch
  - Setzen der gerundeten Variablen $x_e := 1$
  - Anpassung der Bedarfe $f(S) := f(S) - \sum_{e \in \delta(S)} x_e$

9.7 Satz (Gütegarantie des Algorithmus von Jain)

Der Algorithmus von Jain erzeugt eine zulässige Lösung x des allgemeinen Netzwerk Design Problem mit

$$\sum_{e \in A} c_e \leq 2 \cdot OPT(I),$$

d.h. der Algorithmus hat eine Gütegarantie von 2.

Beweis durch Induktion nach der Anzahl der Iterationen ❑

Bemerkungen zum Algorithmus von Jain

- Die LP-Relaxation hat exponentiell viele Ungleichungen. Sie kann dennoch in polynomialer Zeit gelöst werden da
  - Separierung und Optimierung polynomial äquivalent sind (Satz 7.26)
  - das Separierungsproblem für $\sum_{e \in \delta(S)} x_e \geq f(S)$ in polynomialer Zeit durch eine Folge von Min Schnitt

Problemen gelöst werden kann.

- Der Algorithmus ist weniger nützlich in der Praxis, da er eine Folge von LPs lösen muss.
- Es ist offen, ob ein praktischer 2-Approximationsalgorithmus existiert.

10.1 LP ist in NP ∩ coNP

Wichtigste Aussagen dieses Kapitels (alle ohne vollständigen Beweis)

- Lineare Programmierung (LP) liegt in NP ∩ coNP. Daher wurde allgemein vermutet, dass es einen polynomialen Algorithmus für LP geben muss.
- Alle bekannten Varianten des Simplexalgorithmus zeigen jedoch eine exponentielle Worst-Case Laufzeit. Allerdings zeigen die Average Case Analyse und die Smoothed Analysis polynomiale Laufzeit.
- Die Ellipsoidmethode ist das historisch erste Verfahren mit einer polynomialen Worst-Case Laufzeit für LP (Khachiyan 1979). Es hat jedoch keine praktische Bedeutung.
- Innere Punkte Methoden wurden kurz nach der Ellipsoidmethode entwickelt (zuerst von Karmarkar 1984). Sie haben ebenfalls eine polynomiale Worst-Case Laufzeit für LP. Heutige Varianten (Log Barrier, primal-dual) sind dem Simplexalgorithmus in der Praxis ebenbürtig und für sehr große und dünn besetzte Probleme überlegen. Allerdings sind sie eher ungeeignet zum Lösen einer Serie von Optimierungsaufgaben (was für viele Algorithmen der ganzzahligen Optimierung, z. B. Branch and Bound oder Cutting-Plane-Verfahren wichtig ist).

Kodierungslänge eines LP

LP sei gegeben durch

$$\min\ c^T x$$

unter $Ax = b$

$x \geq 0$

mit rationalen Daten A, b, c

Die Kodierungslänge (Größe) von LP bzgl. der Standardkodierung (vgl. ADM I) ist dann

$\langle LP \rangle = \langle A \rangle + \langle b \rangle + \langle c \rangle$

Eine andere, bei Innere Punkte Verfahren benutzte Definition ist

$L := \langle det_{max} \rangle + \langle b_{max} \rangle + \langle c_{max} \rangle + m + n$

mit

$det_{max} := \max \{ |\det A'| : A' \text{ ist quadratische Untermatrix von } A \}$

$b_{max} := \max_i |b_i|$

$c_{max} := \max_j |c_j|$

10.1 Lemma (Kodierungslänge von LP)

$L \leq \langle LP \rangle$

Beweis beruht auf

$|\det A|$ = Volumen des von den Spalten von A aufgespannten Parallelepiped

$\Rightarrow |\det A| \leq \prod_j \|A_j\|$ ❑

10.2 Lemma (Komponenten von Basislösungen sind mit L Bit darstellbar)

Sei x eine Basislösung von LP mit nicht weiter kürzbaren "Hauptnenner"-Darstellung

$$x_B^T = \left(\frac{p_{B(1)}}{q}, \ldots, \frac{p_{B(m)}}{q} \right)$$

Dann gilt $0 \leq p_i < 2^L$ und $1 \leq q < 2^L$

Beweis analog zu Lemma 3.4 ❑

10.3 Lemma (Zielfunktionswerte verschiedener Basislösungen unterscheiden sich genügend)

Seien x, y zulässige Basislösungen von LP mit $c^T x \neq c^T y$.

Dann ist $|c^T x - c^T y| > 1/2^{2L}$

Beweis

Sei p Hauptnenner von x, q Hauptnenner von y

$$\Rightarrow |c^Tx - c^Ty| = \left|\frac{pc^Tx}{p} - \frac{qc^Ty}{q}\right| = \left|\frac{pq(c^Tx - c^Ty)}{pq}\right|$$

$\geq 1/pq$ da $pq(c^Tx - c^Ty) \neq 0$ und ganzzahlig

$> 1/(2^L 2^L)$ wegen Lemma 10.2 ❑

10.4 Korollar (Es reicht, den Zielfunktionswert bis auf Fehler $1/2^{2L}$ genau zu berechnen)

Sei $z := \min\{ c^Tx \mid x \in P \}$ mit $P = \{ x \in \mathbb{R}^n \mid Ax = b, x \geq 0 \}$

Sei $x \in P$ mit $c^Tx \leq z + 1/2^{2L}$

Dann ist jede zulässige Basislösung x^* mit $c^Tx^* \leq c^Tx$ optimal

Beweis

Annahme y ist optimale Basislösung und x^* ist nicht optimal.

Lemma 10.3 => $|c^Tx^* - c^Ty| > 1/2^{2L}$

=> $c^Tx^* > c^Ty + 1/2^{2L} = z + 1/2^{2L} \geq c^Tx \geq c^Tx^*$, Widerspruch ❑

LP ∈ NP ∩ coNP

Hierzu müssen wir LP als Entscheidungsproblem formulieren:

Input: LP und rationale Zahl λ

Frage: Ist $\min\{ c^Tx \mid Ax = b, x \geq 0 \} \leq \lambda$?

10.5 Satz

LP ∈ NP ∩ coNP

Beweis

LP ∈ NP

müssen ein Zertifikat polynomial beschränkter Länge dafür angeben, dass $\min\{ c^Tx \mid Ax = b, x \geq 0 \} \leq \lambda$

Fall 1: LP hat eine optimale Lösung

=> LP hat zulässige Basislösung x' mit $c^Tx' \leq \lambda$

Lemma 10.2 => die Komponenten von x' sind polynomial in L

$Ax' = b, x' \geq 0$ und $c^Tx' \leq \lambda$ kann in polynomialer Zeit (in L) überprüft werden

=> x' ist ein Zertifikat

Fall 2: LP hat eine zulässige Lösung aber eine nach unten unbeschränkte Zielfunktion

=> das duale Programm (D) $\max\{ y^Tb \mid y^TA \leq c^T, y \text{ beliebig} \}$ hat keine zulässige Lösung

Farkas Lemma für (D) => es gibt $x^* \geq 0$ mit $Ax^* = 0$, $c^Tx^* = -1$

- Wähle als Zertifikat
  - eine zulässige Basislösung von LP um Zulässigkeit zu zeigen
  - eine zulässige Basislösung von $\{ Ax = b, x \geq 0, c^T x = -1 \}$ um Unbeschränktheit zu zeigen
- Beide Basislösungen sind wegen Lemma 10.2 polynomial in L
- Fall 3: LP hat keine zulässige Lösung
- liefert keine "Ja"-Antwort => kein Zertifikat nötig

LP ∈ coNP

- müssen ein Zertifikat polynomial beschränkter Länge dafür angeben, dass $\min \{ c^T x \mid Ax = b, x \geq 0 \} > \lambda$
- Fall 1: LP hat eine optimale Lösung
- Dualitätssatz => Frage ist äquivalent zu $\max \{ y^T b \mid y^T A \leq c^T, y \text{ beliebig} \} > \lambda$

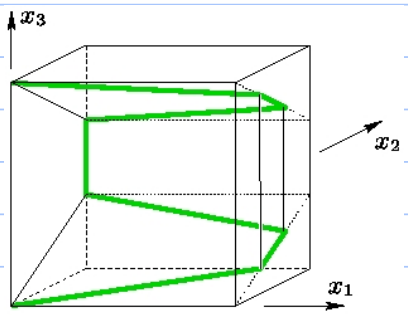
  kann analog zu Fall 1 oben durch eine zulässige Basislösung von $\{ y^T A \leq c^T, y \text{ beliebig} \}$ mit Wert $> \lambda$ zertifiziert werden
- Fall 2: LP hat eine zulässige Lösung aber eine nach unten unbeschränkte Zielfunktion
- => liefert keine "Ja"-Antwort => kein Zertifikat nötig
- Fall 3: LP hat keine zulässige Lösung
- Dann liegt eine "Ja"-Instanz vor, da $\min \{ c^T x \mid Ax = b, x \geq 0 \} = \infty$

Farkas Lemma => es gibt $y \geq 0$ mit $y^T A = 0$, $y^T b = -1$

=> nehme als Zertifikat eine zulässige Basislösung von $\{ y \geq 0 \text{ mit } y^T A = 0, y^T b = -1 \}$ ❑

Worst-Case Laufzeit des Simplexalgorithmus

- Die Worst-Case Laufzeit des Simplexalgorithmus ist exponentiell
- Die Gegenbeispiele sind in der Regel sogenannte Klee-Minty Würfel, d.h. leicht verzerrte Würfel, auf denen der Simplexalgorithmus alle Ecken abläuft, obwohl er schon in einem Schritt fertig sein könnte.



@ http://www.mathematik.de/ger/information/forschungsprojekte/zieglergeometrie/zieglergeometrie.html

Mittlere Laufzeit des Simplexalgorithmus

- Erstes Ergebnis von Borgwardt 1982
  - Variante des Simplexalgorithmus: Schatteneckenalgorithmus

  - Pivotregel basiert auf 2-dimensionaler Projektion des Polyeders
- Verteilungsannahmen
  - b = **1** (o.B.d.A.)

    c und die Zeilen von A sind unabhängige, identisch verteilte Zufallsvektoren, deren Verteilung symmetrisch bzgl. Rotationen um den Nullpunkt ist
  - Erwartete Anzahl von Pivotoperationen $O(n^4 m)$

- Verbesserungen von Haimovich 1983
  - $O(n+m)$ bei gleichem Algorithmus und gleicher Verteilungsannahme

- Einschränkungen
  - Aussage nur für das arithmetische Model mit $O(1)$ pro Operation
  - keine Aussage für den Standardsimplex
  - Verteilungsannahmen erzeugen keine dünn besetzten LPs
  - Bei festem n und wachsendem m geht die Wahrscheinlichkeit für die Existenz einer zulässigen Lösung schnell gegen 0

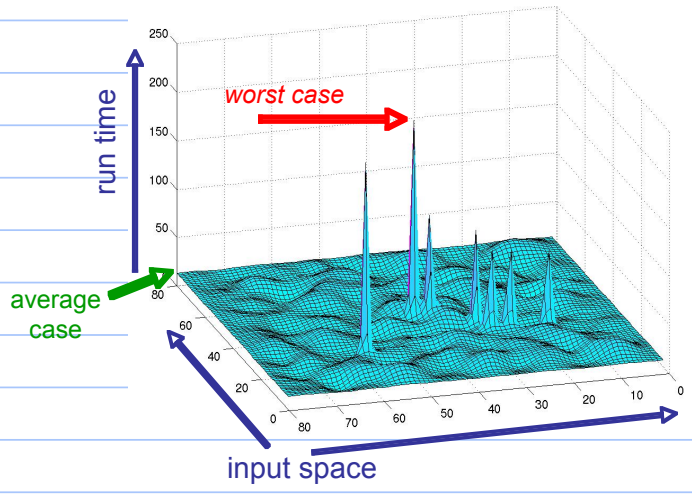
Smoothed Analysis (Geglättete Analyse) des Simplexalgorithmus

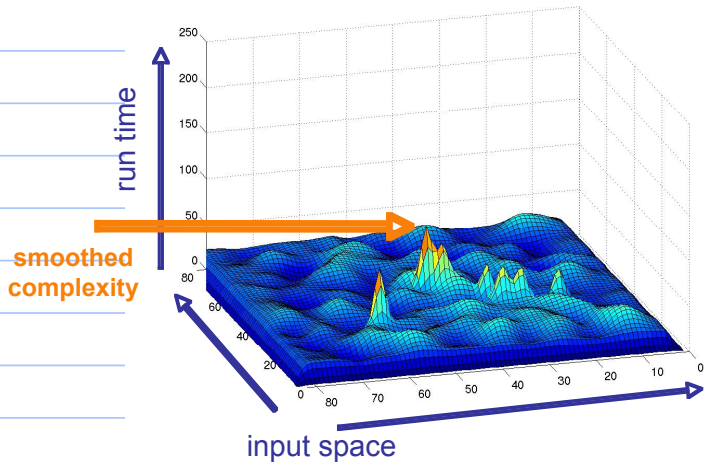
- neues Komplexitätsmodell von Spielman & Teng 2002 eingeführt und auf LP angewendet (95 Seiten Paper)
  - für jede Instanz I betrachte eine Umgebung $N(I)$ mit einer Wahrscheinlichkeitsverteilung über $N(I)$ und berechne $\sup \{ \mathbb{E}_{I^* \in N(I)}[\text{Laufzeit}(I^*)] \mid \text{alle Instanzen } I \}$
  - Spezialfälle:

    Worst Case Analyse: $N(I) = \{I\}$

    Average Case: $N(I)$ = Menge aller Instanzen

    Smoothed Analysis "interpoliert" zwischen diesen Extremen





© Spielman & Teng

- Ergebnis von Spielman & Teng
  - Variante des Simplexalgorithmus: zweistufiger Schatteneckenalgorithmus
  - Verteilungsannahme ist Normalverteilung, bei der die Werte aus A und b, die $\neq 0$ sind, um eine Normalverteilung $N(0,\sigma)$ "perturbiert" werden
  - Umgebung $N(I)$ ist gegeben durch die Standardabweichung σ der Normalverteilung

- Laufzeit in der geglätteten Analyse ist polynomial in n, m, und $1/\sigma$

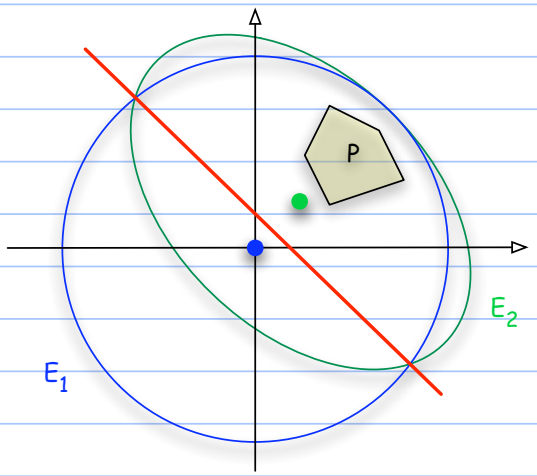
- Einschränkungen
  - keine Aussage für den Standardsimplex
  - Modell erhält die Dünnbesetztheit, aber nicht die Degeneriertheit

- Die geometrische Intuition ist einfach, die technischen Details (und der Nachweis der Polynomialität) sind ...
- **Reduktion der Linearen Optimierung auf Finden eines zulässigen Punktes**
  - eine Möglichkeit: binäre Suche bzgl. d mit Ungleichung $cx \leq d$
  - andere Möglichkeit: Nutzen von Dualität
    - gleichzeitig Bedingungen des primalen und dualen aufstellen

      und Restriktion $c^T x \leq b^T y$

      => einzig zulässiger Punkt ist $(x,y)^T$ mit x primal optimal, y dual optimal

- **Ellipsoidmethode sucht einen zulässigen Punkt in einem Polytop P**
  - Starte mit Kugel E um den Nullpunkt, die P enthält
  - while Volumen von E ist nicht zu klein do // damit noch ein Punkt drin ist
    - if Mittelpunkt x von E ist in P then return x
    - berechne Hyperebene, die x von P separiert, sei H der Halbraum, der P enthält
    - berechne neues Ellipsoid E mit kleinstem Volumen, das $H \cap E$ enthält
  - return "es gibt keine zulässige Lösung"



- Bemerkungen zur Ellipsoidmethode
  - für das neue Ellipsoid kann eine geschlossene Formel angegeben werden (effizienter update)
  - das Volumen schrumpft pro Iteration um einen Faktor $\exp(-1/2n) < 1$
  - "zu kleines" Volumen von P wird bei volldimensionalen Polytopen dadurch realisiert, dass man zu strikten um

$1/2^{2L+1}$ verschobenen Ungleichungen übergeht, so dass P eine Kugel mit Radius $r = 1/2^{2L}$ enthält

=> man kann abbrechen, wenn das Volumen von E unter das dieser Kugel sinkt

- bei niederdimensionalen Polytopen braucht man Zusatzüberlegungen
- Laufzeit insgesamt $O(n^2L)$ Iterationen mit $O(n^4L)$ arithmetischen Operationen mit Zahlen von $O(L)$ Bits
- generelle Info zur Ellipsoidmethode

M. Grötschel, L. Lovász, and A. Schrijver

Geometric Algorithms and Combinatorial Optimization

Springer-Verlag, Berlin, 2nd ed., 1993

Ziel dieses Abschnittes

- Skizze des Innere Punkte Algorithmus von Ye mit Verbesserungen von Freund (beide publiziert in Mathematical Programming 1991)

Ausgangspunkt und generelle Idee

- Gegeben sind ein primales LP und das zugehörige duale in folgender Form

(P) min $z = c^Tx$

unter $Ax = b,\ x \geq 0$

(D) max $w = b^Ty$

unter $A^Ty + s = c,\ s \geq 0$ (Schlupfvariable), y nicht vorzeichenbeschränkt

- Der Algorithmus löst gleichzeitig (P) und (D).
- Er berechnet in jeder Phase eine primale Lösung $x^* > 0$ und duale Schlupfvariablen $s^* > 0$
- Grundidee:
  - Bleibe dem Rand $x_j \geq 0,\ s_j \geq 0$ fern,
    aber mache die Dualitätslücke $c^Tx^* - b^Ty^* = (A^Ty^* + s)^Tx^* - (Ax^*)^Ty^* = x^{*T}s^* > 0$ klein

Die zwei Hauptbestandteile

Bestandteil 1: Skalieren

Seien $x^* > 0$ und $s^* > 0$ gegeben

Die Skalierung ist eine Funktion $\mathbb{R}^n \to \mathbb{R}^n$ mit

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \to x' = \begin{pmatrix} \frac{x_1}{x_1^*} \\ \vdots \\ \frac{x_n}{x_n^*} \end{pmatrix}$$

Beachte:

$$x^* \to \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

Skalierung in Matrixschreibweise:

$$x' = (X^*)^{-1}x \quad \text{mit } X^* = \begin{pmatrix} x_1^* & \dots & 0 \\ & \ddots & \\ 0 & \dots & x_n^* \end{pmatrix}$$

Mit Hilfe der Skalierung lässt sich (P) zunächst umschreiben zu

(P) min $z = c^T X^* x'$

unter $AX^*x' = b,\ x' \geq 0$

Setze $c^* := X^*c,\ A^* := X^*A$

=> (P) kann weiter umgeschrieben werden zu

(P) min $z = c^{*T}x'$

unter $A^*x' = b,\ x' \geq 0$

Entsprechend wird (D) zu

(D) max $w = b^T y$

unter $A^{*T}y + s' = c^*,\ s' \geq 0$ mit

$$s' := X^* \cdot s = \begin{pmatrix} s_1 x_1^* \\ \vdots \\ s_n x_n^* \end{pmatrix}$$

Beachte: $x_j s_j = x_j' s_j'$ => Dualitätslücke bleibt invariant unter Skalierung

=> kann im transformierten Raum weiter arbeiten

Bestandteil 2: Potenzialfunktion

Misst, wie klein die Dualitätslücke ist. Ist eine logarithmic barrier function

$G(x,s) := q \cdot \ln(x^T s) - \sum_j \ln(x_j \cdot s_j)$

mit $q > 0$ geeignet gewählter Parameter

Beachte

$q \cdot \ln(x^T s) \to -\infty$ falls die Lücke $x^T s \to 0$

$- \sum_j \ln(x_j \cdot s_j) \to +\infty$ falls $x_j \to 0$ oder $s_j \to 0$, d.h. dicht am Rand

Frage: wie soll man q wählen?

Als gute Wahl von q erweist sich

$q := n + \sqrt{n}$

Diese Wahl führt zu

$O(\sqrt{n} \cdot L)$ Iterationen

mit L := Kodierungslänge aus Abschnitt 10.1

Abbruchkriterium

Die Potenzialfunktion liefert ein Abbruchkriterium, das auf Lemma 10.2 aufbaut:

Seien x, s primal-dual zulässig mit

$G(x,s) \le -k\sqrt{n}L$ für eine Konstante k

Dann ist $x^T s \le e^{-kL}$

Also: Stoppen sobald

$G(x,s) \le -k\sqrt{n}L$

mit $k = 2$

Beachte: Die Skalierung ändert nicht den Wert von $G(x,s)$

=> man kann im ursprünglichen oder im transformierten Raum rechnen

Der Algorithmus von Ye

Grobstruktur des Algorithmus von Ye

Input

Primal-duales Paar in der Form

(P) min $z = c^T x$

unter $Ax = b,\ x \geq 0$

(D) max $w = b^T y$

unter $A^T y + s = c,\ s \geq 0$ (Schlupfvariable), y nicht vorzeichenbeschränkt

- Output
  - Primal-duales Paar (x,s) mit $G(x,s) \leq -k\sqrt{n}L$
- Methode
  - Initialisierung
    - $i := 0$ // Zähler
    - wähle $x^0 > 0, s^0 > 0$ primal-dual zulässig mit $G(x^0,s^0) = O(\sqrt{n}L)$

      // Idee: Phase I des Simplexalgorithmus modifizieren, so dass $x \sim 2^L,\ s \sim 2^L$
  - Iteration
    - while $G(x^i,s^i) > -2\sqrt{n}L$ do
      - mache primalen Schritt // nur x^i ändern

        oder dualen Schritt // nur s^i ändern

        dies ergibt (x^{i+1}, s^{i+1})
      - $i := i+1$

- Details der Iteration
  - Übersicht
    - Skaliere das momentane Paar $(x^i, s^i) \rightarrow (e, s')$ mit $e = \mathbf{1}$

      $\Rightarrow$ (e, s') ist weit entfernt vom Rand

      der primale oder duale Schritt ergibt dann $(\tilde{x}, \tilde{s})$ und reduziert G

      die Rücktransformation von $(\tilde{x}, \tilde{s})$ in den ursprünglichen Raum ergibt (x^{i+1}, s^{i+1})
  - Haupteigenschaft des primal/dualen Schrittes
    - mache ihn so, dass $G(x^{i+1}, s^{i+1}) - G(x^i, s^i) \leq -7/120 < 0$

      $\Rightarrow$ $G(x^N,s^N) < -2\sqrt{n}L$ nach N Schritten mit

      $$\underbrace{k\sqrt{n}L}_{\approx G(x^0,s^0)} - N\frac{7}{120} \leq 2\sqrt{n}L$$

      $$\Rightarrow N \geq \frac{120}{7}(k+2)\sqrt{n}L = O(\sqrt{n}L)$$

- Berechnung von $(\tilde{x}, \tilde{s})$
  - Um $(\tilde{x}, \tilde{s})$ zu berechnen, betrachten wir den Gradienten von G im Punkt (e, s') nach x:

$$g := \nabla_x G(x,s)|_{(e,s')} = \frac{q}{x^T s} s - \left(\begin{array}{c} \frac{1}{x_1} \\ \vdots \\ \frac{1}{x_n} \end{array} \right) \Bigg|_{(e,s')} = \frac{q}{x^T s'} s' - e$$

  Gehe dann in Richtung $-g$ um G zu verkleinern, aber bleibe zulässig (d.h. $A^* \cdot \tilde{x} = b$).
  - Dazu sei d die Projektion von g auf den Unterraum $\{ x \mid A^* x = 0 \}$

    $\Rightarrow$ $d = (I - A^*(A^* A^{*T})^{-1} A^*) g$ (ohne Beweis)

    Gehe dann in Richtung $-d$

  - Mögliches Problem: $||d||$ ist sehr klein

    $\Rightarrow$ primaler Schritt verkleinert G nicht genügend

    Daher: primaler Schritt für $||d|| \geq 0.4$

    dualer Schritt für $||d|| < 0.4$

- Primaler Schritt

  - Setze $\tilde{x} := e - \frac{1}{4||d||} d, \quad \tilde{s} := s$
  - Nach dem primalen Schritt gilt $\tilde{x} > 0$ und $G(\tilde{x}, \tilde{s}) - G(e, s') \leq -7/120$

- Dualer Schritt
  - hier betrachten wir den Gradienten von G im Punkt (e, s') nach s:

$$h := \nabla_s G(x,s)|_{(e,s')} = \frac{q}{x^T s'} e - \left(\begin{array}{c} \frac{1}{s'_1} \\ \vdots \\ \frac{1}{s'_n} \end{array} \right)$$

    $\Rightarrow$ $h_j = g_j / s_j$ $\Rightarrow$ h und g zeigen in etwa in die gleiche Richtung
  - gehe in die Richtung $-(g-d)$ und setze

    $\tilde{s} := s' - (g-d)\mu$ mit $\mu := e^T s' / q$
  - Nach dem dualen Schritt gilt

$$\tilde{s} = \frac{e^T s'}{q}(d + e), \quad \tilde{x} = x' = e$$

    $\tilde{s} > 0$

$G(\tilde{x}, \tilde{s}) - G(e, s') \leq -1/6$

Analyse des Algorithmus von Ye

Fortschritt pro Iteration

G nimmt in jedem primalem und dualem Schritt um einen konstanten Betrag ab

$\Rightarrow$ $O(\sqrt{n}L)$ Iterationen bis $G(x^i, s^i) < -2\sqrt{n}L$

Laufzeit

Jede Iteration kann in $O(n^3)$ Operationen durchgeführt werden

schwierig ist nur die Berechnung des projizierten Gradienten d

= Lösung des linearen Gleichungssystems $(A*A^{*T})w = A*g$

geht mit Gauß-Elimination in $O(n^3)$ Operationen

=> $O(n^{3.5}L)$ Operationen insgesamt

Problem: Operationen sind nicht exakt möglich

$||d||$ ist im Allgemeinen irrational

=> nur mit fester Bitzahl L rechnen und runden

=> 19/352 statt 7/120 durch Runden bei Berechnung von $||d||$

Die Zahlen dürfen bei der Gauß-Elimination nicht zu groß werden

Nutze $\langle \det B \rangle \leq \langle A \rangle$ für jede quadratische Teilmatrix von A

=> (Cramersche Regel) alle Zahlen in der Gauß-Elimination sind mit L Bits darstellbar

10.6 Satz (Polynomiale Laufzeit des Algorithmus von Ye)

Der Algorithmus von Ye hat eine Laufzeit von $O(n^{3.5}L)$

ohne Beweis ❑