Die lineare Programmierung

Die Standardform der linearen Programmierung

- Für n reellwertige, nichtnegative Variablen $x_1 \geqslant 0, \ldots, x_n \geqslant 0$ erfülle die m linearen Gleichungen

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad \text{für } i = 1, \ldots, m.$$

- Unter allen Lösungsvektoren $x := (x_1, \ldots, x_n)^T$ suche einen Vektor, der die lineare Zielfunktion

$$c^T \cdot x = \sum_{j=1}^{n} c_j x_j$$

minimiert.

Matrix Notation

Wenn

$$A = \left[a_{ij}\right]_{1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n}$$

die $m \times n$-Matrix des Gleichungssystems ist, dann

$$\text{minimiere } c^T \cdot x, \text{ so dass } A \cdot x = b, x \geqslant 0.$$

(1) Wenn $c^T \cdot x$ maximiert werden soll: minimiere $-c^T \cdot x$.

(2) Falls $x \geqslant 0$ nicht gefordert werden soll: Ersetze x durch die Differenz $x_+ - x_-$ für neue Variablen x_-,x_+ mit $x_+ \geqslant 0$ und $x_- \geqslant 0$.

(3) Wie reduziert man die **kanonische Form**

$$\text{minimiere } c^T \cdot x, \text{so dass } A \cdot x \geqslant b, \ x \geqslant 0$$

auf die **Standardform**? Sind beide Formen äquivalent?

Das gewichtete Matching Problem

Das gewichtete Matching Problem

Für einen Graphen $G = (V, E)$ mit Kantengewichten $w_e \geqslant 0$ für $e \in E$ bestimme ein schwerstes Matching $M \subseteq E$.

Keine zwei Kanten in M besizen einen gemeinsamen Endpunkt.

- $0 \leqslant x_e \leqslant 1$. (Wir wollen aber integrale Bedingungen $x_e \in \{0, 1\}$.)
- Damit jeder Knoten $v \in V$ nur Endpunkt **einer** Matchingkante ist:

$$\sum_{w,\{v,w\}\in E} x_{\{v,w\}} \leqslant 1$$

- Die Zielfunktion ist

$$\sum_{e\in E} w_e \cdot x_e$$

Wir zeigen später, dass dieses Programm für **bipartite Graphen** nur integrale Ecken besitzt: Die Ecken entsprechen also Matchings!

Das Flussproblem

Das Flussproblem

- Ein gerichteter Graph $G = (V, E)$ mit Quelle $s \in V$ und Senke $t \in V$ ist gegeben.
- Die Funktion $c : E \to \mathbb{R}_{\geqslant 0}$ weist jeder Kante $e \in E$ ihre maximale Kapazität $c(e) \geqslant 0$ zu.
- Ein Fluss ist ebenfalls eine Funktion $f : E \to \mathbb{R}_{\geqslant 0}$ mit den folgenden Eigenschaften:
  (1) $0 \leqslant f(e) \leqslant c(e)$ für jede Kante $e \in E$: Der Fluss entlang einer Kante darf die Kapazität der Kante nicht übersteigen.
  (2) Für jeden Knoten $v \in V \setminus \{s, t\}$ gilt Flusserhaltung:

$$\sum_{(u,v)\in E} f(u, v) = \sum_{(v,u)\in E} f(v, u)$$

- Das Ziel des Flussproblems ist die Konstruktion eines maximalen Flusses von der Quelle s zur Senke t.

Das Flussproblem als lineares Programm

- Einhaltung der Kapazitätsschranke: $0 \leqslant f_e \leqslant c(e)$.
- Flusserhaltung für jeden Knoten $v \in V \setminus \{s, t\}$:

$$\sum_{(v,w)\in E} f_{(v,w)} - \sum_{(w,v)\in E} f_{(w,v)} = 0.$$

- Maximiere den Fluss, der s verlässt ohne zurückzufliessen:

$$\text{maximiere} \quad \sum_{(s,v)\in E} f_{(s,v)} - \sum_{(v,s)\in E} f_{(v,s)}.$$

Wir zeigen später: Wenn alle Kapazitäten ganzzahlig sind, dann ist auch ein maximaler Fluss ganzzahlig.

Die geometrische Sicht: Polytope, Polyeder, Halbräume und Ecken

Fundamentale Konzepte

(a) Ein Vektor x mit $Ax = b$ und $x \geqslant 0$ heißt eine **Lösung**.

(b) Ein lineares Programm heißt **lösbar**, wenn Lösungen existieren, ansonsten ist das Programm **unlösbar**.

(c) Wir bezeichnen die Lösungsmenge mit $L(A, b)$, wobei

$$L(A, b) := \{x \in \mathbb{R}^n \mid Ax = b,\ x \geqslant 0\}.$$

(d) Eine Lösung $x \in L(A, b)$ ist **optimal**, falls

$$c^T \cdot x = \inf\{c^T \cdot x \mid x \in L(A, b)\}.$$

(d) Eine Lösung $x^* \in L(A, b)$ ist genau dann eine **Ecke** von $L(A, b)$, wenn kein Vektor $y \neq 0$ mit $x^* + y \in L(A, b)$ und $x^* - y \in L(A, b)$ existiert.

Die geometrische Sichtweise: Polytop und Polyeder

Die Lösungsmenge $L(A, b)$ wird auch **Polyeder** genannt.
Wenn $L(A, b)$ beschränkt ist, dann ist $L(A, b)$ ein **Polytop**.

Die geometrische Sichtweise für die kanonische Form

$$L^*(A, b) = \{x \mid Ax \geqslant b, x \geqslant 0\}.$$

- Eine Ungleichung $\sum_{j=1}^{n} A[i, j] \cdot x_j \geqslant b_i$ definiert einen **Halbraum**, der den $\mathbb{R}^n$ in zwei Teile zerschneidet.
- Ein Polyeder ist ein Durchschnitt von Halbräumen.

Die geometrische Sichtweise: Seitenflächen

- Die **Dimension** eines Polyeders P ist definiert als die Dimension des kleinsten affinen Raums, der P enthält.
- Liegt ein Polyeder P *vollständig auf einer Seite einer Hyperebene* H, dann wird der Durchschnitt $P \cap H$ eine **Seitenfläche** genannt.

- Eine **Ecke** x ist eine 0-dimensionale Seitenfläche: Der Durchschnitt $P \cap H$ besteht nur aus x.
- Eine **Facette** F ist eine Seitenfläche im „wirklichen Sinn“: Der Durchschnitt $F = P \cap H$ ist ein $n-1$-dimensionales Polyeder.
- Eine Ecke x heißt **entartet**, wenn die Anzahl der Facetten, die x enthalten, größer ist als die Dimension von P.

  Die Spitze einer 3-dimensionalen Pyramide mit quadratischer Grundfläche ist entartet.

Die geometrische Sichtweise: Konvexität

Eine Menge $X \subseteq \mathbb{R}^n$ ist **konvex**, wenn für je zwei Punkte $u, v \in X$ auch die u, v verbindende Gerade vollständig in X liegt.

- Ein Polyeder P ist stets konvex.
- Die Menge aller Konvexkombinationen der Punkte $x_1, \ldots, x_N \in \mathbb{R}^n$ ist die Menge

$$\{ \sum_{i=1}^{N} \lambda_i \cdot x_i \mid \lambda_1, \ldots, \lambda_N \geqslant 0, \sum_{i=1}^{N} \lambda_i = 1. \}.$$

- Jedes Polytop stimmt mit der Menge aller Konvexkombination seiner Ecken überein.

Es gibt immer optimale Ecken

Es gibt stets eine optimale Ecke (1/3)

Die Lösungsmenge $L(A, b)$ sei nichtleer und es gelte

$$-\infty < \inf \{c^T \cdot x \mid x \in L(A, b)\} < \infty.$$

Dann gibt es eine Ecke, die optimal ist.

- O.B.d.A. ist $c \neq 0$: Falls $c = 0$ ist nur zu zeigen, dass eine Ecke existiert.
- Sei $x \in L(A, b)$ eine optimale Lösung, aber keine Ecke. Wir konstruieren aus x eine mindestens so gute Ecke.
  - Es gibt $y \neq 0$ mit $x \pm y \in L(A, b)$.
  - Also ist $x \pm y \geqslant 0$ und aus $Ax + Ay = b$ und $Ax = b$ folgt $Ay = 0$.
  - O.B.d.A. $c^T \cdot y \leqslant 0$.

    Falls $c^T \cdot y = 0$, wählen wir zwischen y und $-y$, so dass der gewählte Vektor mindestens eine negative Komponente besitzt.

Fall 1: Es gibt einen Eintrag j mit $y_j < 0$.

Wähle $\lambda > 0$ maximal, so dass $x + \lambda y \geqslant 0$.

- $\lambda > 0$: Der Vektor $x + \lambda y$ hat im Vergleich zu x mindestens eine Null-Komponente mehr:

  Wenn $x_i = 0$, dann muss $y_i = 0$ gelten, da $x \pm y \in L(A, b)$.
- Setze $x^{\text{neu}} := x + \lambda y \geqslant 0$.
- Zeige $A \cdot x^{\text{neu}} = b$: Wegen $x^{\text{neu}} \geqslant 0$, folgt dann $x^{\text{neu}} \in L(A, b)$.

$$A \cdot x^{\text{neu}} = A(x + \lambda y) = Ax + \lambda \cdot \underbrace{(Ay)}_{=0} = Ax = b.$$

x^{neu} hat im Vergleich zu x mindestens eine Null-Komponente mehr.

Es gibt stets eine optimale Ecke (3/3)

Fall 2: Für alle j ist $y_j \geqslant 0$.

Für jedes $\lambda \geqslant 0$ ist also $x + \lambda y \geqslant x \geqslant 0$.

- Nach Konstruktion von y gilt also $c^T y < 0$.
- Für jedes $\lambda \geqslant 0$ ist $x + \lambda y \in L(A, b)$.
  - Für jedes λ ist $A(x + \lambda y) = Ax + \lambda \cdot \underbrace{(Ay)}_{=0} = Ax = b$
  - und $x + \lambda y \geqslant x \geqslant 0$.
- Es ist $c^T y < 0$ und $c^T(x + \lambda y) = c^T x + \lambda \cdot c^T y$.

  Das Optimum ist $-\infty$, im Widerspruch zur Annahme.

Fall 2 tritt also nicht auf: Die wiederholte Anwendung von Fall 1 führt auf eine Ecke oder auf den Nullvektor, der aber auch eine Ecke ist.

Ecken: Eine algebraische Charakterisierung (1/2)

- $x \in L(A, b)$ sei eine Lösung mit $B = \{j \mid x_j > 0\}$.
- A besitze die Spalten $A_1, \ldots, A_n$ und die Matrix A_B bestehe aus den Spalten A_i für $i \in B$.
- Wenn B nicht-leer ist, dann

$$x \text{ ist eine Ecke} \quad \Leftrightarrow \quad \text{die Spalten von } A_B \text{ sind linear unabhängig.}$$

x sei eine Ecke. Wenn die Spalten von A_B linear abhängig sind,

- dann gibt es einen Vektor $y_B \neq 0$ mit $A_B \cdot y_B = 0$.
- Ergänze y_B durch Null-Komponenten zu $y \in \mathbb{R}^n$ mit $Ay = 0$.
  - Es ist $y_i = x_i = 0$ für $i \notin B$ und $x_i > 0$ für $i \in B$.
  - Also gibt es $\lambda > 0$ mit $x \pm \lambda y \geqslant 0$.
- Aber $A(x \pm \lambda y) = Ax \pm \lambda \cdot (Ay) = Ax = b$.

$x \pm \lambda y \in L(A, b)$ und x ist keine Ecke.

Ecken: Eine algebraische Charakterisierung (2/2)

Und wenn x keine Ecke ist?

Dann gibt es ein $y \neq 0$ mit $x \pm y \in L(A, b)$.

- Aus $Ax + Ay = b$, $Ax - Ay = b$ folgt $Ay = 0$.
- Es ist $x + y \geqslant 0$, $x - y \geqslant 0$.
  - Wenn $x_i = 0$ (oder äquivalent, wenn $i \notin B$), dann ist $y_i = 0$.
  - Also ist $A_B \cdot y_B = A \cdot y = 0$ für die Einschränkung y_B von y auf die Komponenten in B.
  - Es ist also insbesondere $y_B \neq 0$, da $y \neq 0$.
  - $0 = A_B \cdot y_B$ und A_B hat wegen $y_B \neq 0$ linear abhängige Spalten.

Wenn x keine Ecke ist, dann hat A_B linear abhängige Spalten

Der Simplex Algorithmus

Der Simplex Algorithmus: Ein Überblick

Wir machen die folgenden Annahmen:

- Die Zielfunktion $c^T \cdot x$ ist für $x \in L(A, b)$ zu minimieren.
- A hat m Zeilen und n Spalten.
- Es gilt $\text{Rang}(A) = m$.
  - Wenn $\text{Rang}(A) < m$ ist, dann sind Restriktionen überflüssig oder es gibt überhaupt keine Lösung.
  - Entferne gegebenenfalls überflüssige Zeilen aus der Matrix A und die entsprechenden Einträge im Vektor b.
- Der Lösungsraum ist nichtleer.

Der Algorithmus

(1) Wähle eine beliebige Ecke x des Lösungspolyeders.

(2) Bestimme eine benachbarte Ecke x^{neu} mit niedrigerem Zielwert.

(3) Falls keine existiert, stoppe mit der Ausgabe x. Ansonsten setze $x := x^{\text{neu}}$ und gehe zu Schritt (2).

Basislösungen

Wie findet man benachbarte Ecken?

x sei eine Ecke.

- Die Spalten der Matrix A_B für $B = \{i \mid x_i > 0\}$ sind linear unabhängig.
- Wenn $|B| = m$: Es gilt $b = A \cdot x = A_B \cdot x_B$ und $(A_B)^{-1} b = x_B > 0$ folgt: x ist eine nicht-entartete Ecke.
- Wenn $|B| < m$: Ergänze A_B durch $m - |B|$ linear unabhängige Spalten von Matrix A zu einer regulären Matrix $A_{B'}$:
  x ist, als Schnittpunkt von mehr als n Hyperebenen, entartet.

$B \subseteq \{1, \ldots, n\}$ mit $|B| = m$ heißt **Basis**, falls

- A_B regulär ist und
- $A_B^{-1} b \geqslant 0$ gilt.

$x = (x_B, x_N)$ mit $x_B = A_B^{-1} b$ und $x_N = 0$ ist die **Basislösung** zu B.

Ein Schritt von Simplex (1/2)

- Für eine Lösung x zur Basis B bezeichne $x_B, x_N, c_B, c_N, A_B, A_N$ die Einschränkungen auf Basis- bzw. Nicht-Basis-Komponenten.
- Selbst wenn die Lösung x keine Basislösung zur Basis B ist:
  - $c^T \cdot x = c_B^T \cdot x_B + c_N^T \cdot x_N$,
  - $Ax = A_B \cdot x_B + A_N \cdot x_N$,
  - $x_B = A_B^{-1} \cdot b - A_B^{-1} \cdot A_N \cdot x_N$,
  - $x_B, x_N \geqslant 0$.

Als Konsequenz

$$\begin{aligned} c^T \cdot x &= c_B^T \cdot x_B + c_N^T \cdot x_N \\ &= c_B^T \cdot \left(A_B^{-1} \cdot b - A_B^{-1} \cdot A_N \cdot x_N \right) + c_N^T \cdot x_N \\ &= \underbrace{c_B^T \cdot A_B^{-1} \cdot b}_{\text{aktueller Zielwert}} + \underbrace{\left(c_N^T - c_B^T \cdot A_B^{-1} \cdot A_N \right)}_{=\tilde{c}_N} \cdot x_N. \end{aligned}$$

Ein Schritt von Simplex (2/2)

Für Basis B und eine beliebige Lösung x ist

$$c^T \cdot x = \underbrace{c_B^T \cdot A_B^{-1} \cdot b}_{\text{aktueller Zielwert}} + \underbrace{\left(c_N^T - c_B^T \cdot A_B^{-1} \cdot A_N\right)}_{=\tilde{c}_N} \cdot x_N.$$

- **Fall 1:** Es ist $\tilde{c}_j \geqslant 0$ für alle $j \in N$. Für die Basislösung x^* zur Basis B:
  - $c^T \cdot x^* = c_B^T \cdot A_B^{-1} \cdot b \leqslant c_B^T \cdot A_B^{-1} \cdot b + \tilde{c}_N \cdot x_N$
  - Die Basislösung x^* zu B ist bereits optimal!
- **Fall 2:** Es gibt ein $j \in N$ mit $\tilde{c}_j < 0$.
  - Wir erhöhen nur die Nichtbasisvariable x_j^* der Basislösung x^* von 0 auf $\lambda \in \mathbb{R}$: $x_N^*(\lambda) := (0, \cdots 0, \lambda, 0, \cdots, 0)^T$.
  - Wir passen den Basisteil an: $x_B^*(\lambda) := A_B^{-1} \cdot b - A_B^{-1} \cdot A_N \cdot x_N^*(\lambda)$.
    Ist die Ecke x^* nicht entartet, dann gibt es $\lambda > 0$ mit $x_B^*(\lambda) \geqslant 0$, denn $x_B^* = A_B^{-1} \cdot b > 0$.
  - $A_B \cdot x_B^*(\lambda) + A_N \cdot x_N^*(\lambda) = b - A_N \cdot x_N^*(\lambda) + A_N \cdot x_N^*(\lambda) = b$.

Wähle λ maximal, so dass $x^*(\lambda) := (x_B^*(\lambda), x_N^*(\lambda)) \geqslant 0$.

Der Simplex Algorithmus im Detail

(1) Gegeben ist eine $m \times n$-Matrix A mit $\text{Rang}(A) = m$.
Wir nehmen $L(A, b) \neq \emptyset$ an.

(2) Beginne an einer Ecke x^* mit Basis B.

(3) Sei $j \in N = \{1, \ldots, n\} \setminus B$ minimal mit $\tilde{c}_j < 0$.
Wenn es kein solches j gibt, stoppe mit Ausgabe x^*.
Kommentar: Diese Wahl von j schließt Cycling aus. Existiert kein solches j, haben wir eine optimale Lösung gefunden.

(4) Wähle λ maximal mit $x^*(\lambda) \geqslant 0$.

(4a) Halte, wenn $\lambda = \infty$: Wir erhalten das Infimum $-\infty$.
(4b) Wenn $\lambda > 0$, dann gibt es $k \in B$ mit $x^*_k > x^*(\lambda)_k = 0$.
Setze $B = (B \setminus \{k\}) \cup \{j\}$ und $x^* = x^*(\lambda)$.
(4c) Wenn $\lambda = 0$, wähle $k \in B$ minimal mit $x^*_k = 0$.
Setze $B = (B \setminus \{k\}) \cup \{j\}$.

(5) Gehe zu Schritt (3).

Korrektheit

Wir wissen:

- Wenn $\tilde{c}_j \geqslant 0$ für alle $j \in N$, dann ist x^* optimal.
- Wenn es $j \in N$ mit $\tilde{c}_j < 0$ gibt:
  - Ist die Ecke nicht entartet, dann ist $x^*(\lambda)$ eine bessere Nachbarecke:
    - ★ Der Zielwert wird um $\lambda \cdot \tilde{c}_j$ gesenkt.
    - ★ $(B \setminus \{k\}) \cup \{j\}$ ist wieder eine Basis: Warum?
  - Ist die Ecke entartet, dann ist $\lambda = 0$ möglich.
    - ★ Man kann zeigen: Die Anti-Cycling Regeln der Schritte (3) und (4c) werden kurz über lang eine bessere Nachbarecke finden.

In Schritt (2) setzen wir voraus, dass wir eine Ecke x^* kennen.
Aber woher?

Zwei-Phasen Simplex

Führe Hilfsvariablen r ein und löse

$$\text{minimiere } \sum_i r_i \text{ so dass } Ax + r = b, \quad x, r \geqslant 0$$

mit dem Simplex-Algorithmus.

- $(x, r) = (0, b)$ ist eine Basislösung.
  Multipliziere Zeilen von A und Komponenten von b gegebenenfalls mit -1, um $b \geqslant 0$ zu garantieren.
- Das Programm hat genau dann das Optimum 0, also $r = 0$, wenn das Ausgangsproblem lösbar ist.
- Aus der optimalen Ecke $(x, r) = (x_0, 0)$ des Programms erhalten wir die Ecke x_0 zum Ausgangsproblem.

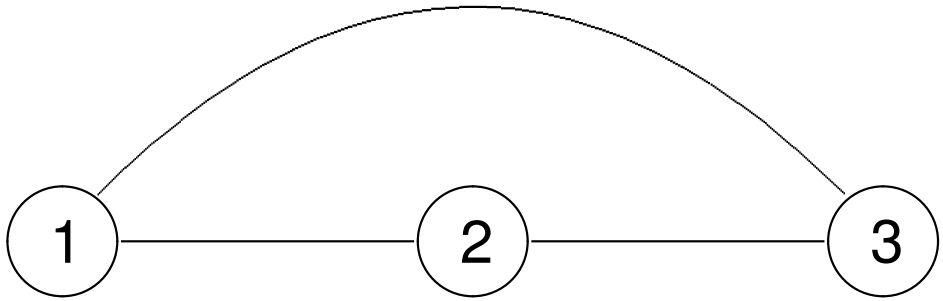
Laufzeitanalyse

- In jeder Iteration müssen wir Matrizen invertieren und multiplizieren.
- Die Laufzeit ist bis auf einen polynomiellen Faktor durch die Anzahl der Ecken bestimmt.
  - Da jede Ecke einer Basis, also einer m-elementigen Teilmenge von $\{1, \ldots, n\}$ entspricht, gibt es höchstens $\binom{n}{m} \leqslant 2^n$ Ecken.
  - Der n-dimensionale Würfel $W_n = \{x \in \mathbb{R}^n \mid 0 \leqslant x_i \leqslant 1\}$ hat genau 2^n Ecken.
- Der Simplex-Algorithmus löst ein lineares Programm mit n Variablen und $m \leqslant n$ Restriktionen in Zeit $O(\text{poly}(n) \cdot 2^n)$.
  - Exponentielles Worst-Case Verhalten tritt auf: Wir beschreiben später effiziente Interior Point Verfahren.
  - Für „in der Praxis auftretende Probleme“ ist das Laufzeitverhalten aber gut.

Wann haben alle Ecken ganzzahlige Komponenten?

- Im Matching Problem für bipartite Graphen und im Flussproblem für ganzzahlige Kapazitäten sind alle Ecken ganzzahlig.
- Warum ist das so?

- Das Matching Problem für allgemeine Graphen ist „bösartig“: Der Graph



  besitzt ein maximales Matching aus genau einer Kante.
- In der Formulierung als lineares Programm erhalten wir die Lösung $x = (x_{\{1,2\}}, x_{\{1,3\}}, x_{\{2,3\}}) = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ mit Wert $\frac{3}{2}$.

Unter welchen Eigenschaften an A sind alle Ecken ganzzahlig?

Vollständige Unimodularität

- Eine Matrix A heißt vollständig unimodular, falls $\det B \in \{-1, 0, 1\}$ für jede quadratische Teilmatrix B von A gilt.
- Falls die Matrix A vollständig unimodular und b ein ganzzahliger Vektor ist, haben die Ecken der Polyeder $\{x \in \mathbb{R}^n \mid Ax \geqslant b\}$ und $\{x \in \mathbb{R}^n \mid Ax = b,\ x \geqslant 0\}$ nur ganzzahlige Komponenten.

- x ist eine Ecke von $L(A, b)$, wenn die Spalten zu $B = \{i \mid x_i > 0\}$ linear unabhängig sind.

  Wenn A genau m Zeilen hat, dann füge $m - |B|$ linear unabängige Spalten zu B hinzu: Dann ist $|B| = m$ und $A_B \cdot x_B = b$, $x_N = 0$.
- Die Cramersche Regel für die Lösung von Gleichungssystemen:

$$\text{Für jedes } i \in B \text{ ist } \quad (x_B)_i = \frac{\det(A_B^i)}{\det(A_B)}.$$

  A_B^i: Ersetze die $i$te Spalte von A_B durch b.
- Aber $\det(A_B) = \pm 1$: x hat nur ganzzahlige Komponenten.

Wann ist eine Matrix A vollständig unimodular?

- Eine vollständig unimodulare Matrix hat nur Einträge aus $\{-1, 0, 1\}$, da jeder Eintrag eine quadratische Teilmatrix ist.
- **Übungsaufgabe**: Inzidenzmatrizen für **gerichtete Graphen** und **ungerichtete bipartite Graphen** sind vollständig unimodular:
  - Für einem gerichteten Graphen $G = (V, E)$ ist $A = (a_{v,e})_{v \in V, e \in E}$ die Inzidenzmatrix von G, falls

$$a_{v,e} = \begin{cases} +1 & \text{falls } e = (v, w) \text{ für ein } w \in V \\ -1 & \text{falls } e = (w, v) \text{ für ein } w \in V \\ 0 & \text{sonst.} \end{cases}$$

  - Für einen ungerichteten Graphen $G = (V, E)$ ist $a_{v,e} = 1$ genau dann, wenn $v \in e$.

Und die Konsequenzen

- Das Polytop des bipartiten Matchings Problem besitzt nur ganzahlige Ecken,
  - denn die Matrix des Ungleichungssystems ist (fast) die Inzidenzmatrix eines ungerichteten bipartititen Graphen.
- Das Polytop des Flussproblems für ganzzahlige Kapazitäten besitzt nur ganzzahlige Ecken,
  - denn die Matrix des Ungleichungssystems ist (fast) die Inzidenzmatrix eines gerichteten Graphen.

Es gibt schnellere kombinatorische Algorithmen für das (allgemeine) Matching- wie auch für das Flußproblem

Das Farkas Lemma

- Wenn ein lineares Gleichungssystem $Ax = b$ keine Lösung hat, ist eine Zeile von den anderen linear abhängig,
- diese Abhängigkeit wird vom Vektor b nicht „eingehalten“:

  Es gibt y mit $y^T \cdot A = 0$ und $y^T \cdot b \neq 0$.
- Gibt es eine ähnliche äquivalente Eigenschaft, um festzustellen, dass ein Polyeder leer ist?

Genau eine der folgenden Aussagen ist wahr:

(a) Es gibt ein x mit $Ax = b, x \geqslant 0$.

(b) Es gibt ein y mit $y^T \cdot A \geqslant 0$ und $y^T \cdot b < 0$.

Und ebenso:

(a) Es gibt ein x mit $Ax \leqslant b$.

(b) Es gibt ein $y \geqslant 0$ mit $y^T \cdot A = 0$ und $y^T \cdot b < 0$.

Das Farkas Lemma: Eine geometrische Interpretation

- Die Spaltenvektoren $A_1, \ldots, A_n$ der Matrix A erzeugen den Kegel

$$\text{Kegel}(A) = \{\sum_{i=1}^{n} x_i A_i \mid x_1, \ldots, x_n \geqslant 0\} = \{Ax \mid x \geqslant 0\}.$$

- Das Farkas Lemma:
  - Entweder liegt der Vektor b in Kegel(A)
  - oder eine Hyperebene $H_y := \{x \mid y^T \cdot x = 0\}$ trennt b und Kegel(A).

Die beiden Eigenschaften schließen sich aus.

Angenommen, es gibt x und y mit

- $Ax = b, x \geqslant 0$ sowie
- $y^T \cdot A \geqslant 0$ und $y^T \cdot b < 0$.

- Wir erhalten

$$y^T \cdot (A \cdot x) = y^T \cdot b < 0.$$

- Aber $y^T \cdot A \geqslant 0$ und $x \geqslant 0$: Wir erhalten den Widerspruch

$$(y^T \cdot A) \cdot x \geqslant 0.$$

Angenommen, es gibt kein $x \geqslant 0$ mit $Ax = b$

Die Menge $K := \{Ax \mid x \geqslant 0\}$ ist abgeschlossen und konvex.

- Nach Voraussetzung ist $b \notin K$. Sei p der Punkt in K mit kürzestem Abstand von b.
- Für alle $z \in K$ gilt $(z - p)^T \cdot (b - p) \leqslant 0$. Warum?
  - Für je zwei Vektoren v, w gilt $\cos \angle(v, w) = \frac{v^T \cdot w}{||v|| \cdot ||w||}$.
  - Der Winkel zwischen $z - p$ und $b - p$ liegt zwischen 90° und 270°.
- Setze $y = p - b$. Als Konsequenz $(Ax - p)^T \cdot (-y) \leqslant 0$.
- Es gilt $y^T A \geqslant 0$. Warum?
  - $p \in K$: Es gibt $w \geqslant 0$ mit $p = A \cdot w$.
  - Wenn $x \geqslant 0$: $0 \leqslant (Ax - p)^T \cdot y = (Ax - Aw)^T \cdot y = (A(x - w))^T \cdot y$.
  - Für $x := w + e_i$ folgt

$$0 \leqslant (Ae_i)^T \cdot y \Rightarrow (y^T \cdot A)_i \geqslant 0$$

und das war zu zeigen.

Warum gilt $y^T \cdot b < 0$?

Wir wissen: $0 \leqslant (Ax - p)^T \cdot y$.

- Für $x = 0$ folgt $0 \leqslant -p^T \cdot y$ oder äquivalent $y^T \cdot p \leqslant 0$.
- $y = p - b$:
  - Es gilt $y^T \cdot b = y^T \cdot (p - y) = y^T \cdot p - y^T \cdot y$ und
  - $y \neq 0$ (denn $b \notin K$ und $p \in K$) und deshalb $y^T \cdot y > 0$.
- Als Konsequenz

$$y^T \cdot b = \underbrace{y^T \cdot p}_{\leqslant 0} - \underbrace{y^T \cdot y}_{>0} < 0.$$

Primales und duales Programm

(P) minimiere $c^T \cdot x$, so dass $A \cdot x = b$ und $x \geqslant 0$

(D) maximiere $y^T \cdot b$, so dass $y^T \cdot A \leqslant c$.

- (P) ist das primale und (D) das zugehörige duale Programm.
- Es gilt die schwache Dualität: $\max_D \leqslant \min_P$
  1. Wähle einen Vektor y und summiere die Gleichungen gemäß y, d.h. berechne $y^T \cdot (A \cdot x) = y^T \cdot b$.
  2. Wenn wir $y^T \cdot A \leqslant c$ fordern, d.h. wenn wir fordern, dass y eine Lösung des dualen Programms ist, dann folgt wegen $x \geqslant 0$

$$y^T \cdot b = y^T \cdot A \cdot x \leqslant c^T \cdot x$$

  3. und $\max_D \leqslant \min_P$ als Konsequenz.

Das Dualitätstheorem

Sei x eine Lösung des primalen Problems und y eine Lösung des dualen Problems, sowie x_{opt} und y_{opt} endliche, optimale Lösungen.

(a) Schwache Dualität: $c^T \cdot x \geqslant b^T \cdot y$

(b) Starke Dualität: $c^T \cdot x_{\text{opt}} = b^T \cdot y_{\text{opt}}$.

- Zeige $c_{\text{opt}} := c^T \cdot x_{\text{opt}} \leqslant b^T \cdot y_{\text{opt}}$.
- Also zeige: Es gibt y mit $y^T \cdot A \leqslant c$ und $c_{\text{opt}} \leqslant b^T \cdot y$.
- Setze als Vorbereitung für das Farkas Lemma

$$\overline{A} := \begin{pmatrix} A^T \\ -b^T \end{pmatrix} \quad \text{und} \quad \overline{b} := \begin{pmatrix} c \\ -c_{\text{opt}} \end{pmatrix}$$

Wenn die starke Dualität nicht gilt: Es gibt kein y mit $\overline{A} \cdot y \leqslant \overline{b}$.

Die Anwendung des Farkas Lemma

Wenn es kein y mit $\overline{A} \cdot y \leqslant \overline{b}$ gibt, dann gibt es $(z, \lambda) \geqslant 0$ mit

$$(z, \lambda)^T \cdot \overline{A} = 0 \text{ und } (z, \lambda)^T \cdot \overline{b} < 0.$$

- Was bedeutet das?
  - $(z, \lambda)^T \cdot \begin{pmatrix} A^T \\ -b^T \end{pmatrix} = 0$ und damit $Az = \lambda b$.
  - $(z, \lambda)^T \cdot \overline{b} = z^T \cdot c - c_{\text{opt}} \cdot \lambda < 0$ und damit $z^T \cdot c < c_{\text{opt}} \cdot \lambda$.
- **Fall 1:** $\lambda = 0$:
  - Dann ist $Az = 0$, $z \geqslant 0$ und $z^T \cdot c < 0$.
  - x_{opt} ist nicht optimal: $x_{\text{opt}} + \mu \cdot z$ ist für jedes $\mu \geqslant 0$ eine Lösung.
- **Fall 2:** $\lambda > 0$. Setze $x = \frac{z}{\lambda}$:
  - $A \cdot x = \frac{1}{\lambda} \cdot (A \cdot z) = \frac{\lambda b}{\lambda} = b$ und
  - $c^T \cdot x = c^T \cdot \frac{z}{\lambda} = \frac{1}{\lambda} \cdot \left(c^T \cdot z\right) < \frac{c_{\text{opt}} \cdot \lambda}{\lambda} = c_{\text{opt}}$.
  - x_{opt} ist auch diesmal nicht optimal, denn x ist besser.

Primale, duale Programme: Die allgemeinen Regeln

Das primale Programm

$$(P) \text{ minimiere } c_1^T \cdot x_1 + c_2^T \cdot x_2, \text{ so dass } \quad A_1 \cdot x_1 = b_1 \text{ und } x_1 \geqslant 0, \quad A_2 \cdot x_2 \geqslant b_2 \text{ und } x_2 \geqslant 0$$

besitzt das duale Programm

$$(D) \text{ maximiere } b_1^T \cdot y_1 + b_2^T \cdot y_2, \text{ so dass } \quad y_1^T \cdot A_1 \leqslant c_1, \quad y_2^T \cdot A_2 \leqslant c_2 \text{ und } y_2 \geqslant 0.$$

- Primale Ungleichung $\Leftrightarrow$ nicht-negative duale Variable.
- Primale Gleichung $\Leftrightarrow$ vorzeichenfreie duale Variable.
- Primale nicht-negative Variable $\Leftrightarrow$ duale Ungleichung.
- Primale vorzeichenfreie Variable $\Leftrightarrow$ duale Gleichung.

Komplementäre Slackness: Die Dualitätslücke

Wir führen Slackvariablen im dualen Problem ein:

$$(P) \quad \text{minimiere } c^T \cdot x, \text{ so dass } A \cdot x = b \text{ und } x \geqslant 0$$
$$(D) \quad \text{maximiere } y^T \cdot b, \text{ so dass } y^T \cdot A + s = c \text{ und } s \geqslant 0.$$

Die *Dualitätslücke* gibt den Abstand zwischen der Lösung x des primalen und der Lösung (y, s) des dualen Problems an:

$$\begin{aligned} c^T \cdot x - b^T \cdot y &= c^T \cdot x - (Ax)^T \cdot y = x^T \cdot c - x^T \cdot A^T \cdot y \\ &= x^T \cdot \left(c - A^T \cdot y\right) = x^T \cdot s. \end{aligned}$$

x und y sind genau dann optimale Lösungen, wenn:

- $x^T \cdot s = 0$,
- für alle i ist $x_i \cdot s_i = 0$ oder
- wenn aus $s_i > 0$ stets $x_i = 0$ folgt.

Komplementäre Slackness

$$(P) \quad \text{minimiere } c^T \cdot x, \text{ so dass } A \cdot x \geqslant b \text{ und } x \geqslant 0$$
$$(D) \quad \text{maximiere } y^T \cdot b, \text{ so dass } y^T \cdot A \leqslant c \text{ und } y \geqslant 0$$

x und y seien Lösungen für das primale Problem (P) und das duale Problem (D). Dann sind die folgenden Aussagen äquivalent:

(a) x und y sind optimal.

(b) Es gelten primale und duale komplementäre Slackness:
- Primale komplementäre Slackness: Für alle i ist entweder $x_i = 0$ oder die x_i zugeordnete duale Ungleichung ist exakt erfüllt.
- Duale komplementäre Slackness: Für alle j ist entweder $y_j = 0$ oder die y_j zugeordnete primale Ungleichung ist exakt erfüllt.

Primal-duale Algorithmen

Es gelte $c \geqslant 0$.

(1) Beginne mit $x = 0$ und der dualen Lösung $y = 0$.
Die folgende Invariante gilt somit anfänglich:
  (a) y ist eine Lösung von (D).
  (b) Die primale komplementäre Slackness Bedingung gilt.

(2) Erfülle duale Bedingungen exakt:
  - Erhöhe eine oder mehrere Komponenten von y bis eine erste duale Ungleichung exakt erfüllt wird.
  - Erhöhe die zu den exakt erfüllten dualen Ungleichungen gehörenden x-Komponenten:

    Die Modifikation von x soll neue primale Ungleichungen erfüllen.

(3) Wiederhole Schritt (2), bis alle primalen Ungleichungen erfüllt sind.

Am Ende ist x eine Lösung. Wenn $c^T x \leqslant \rho \cdot b^T y$, dann ist x eine ρ-approximative Lösung, denn $b^T y \leqslant \mathrm{opt}_P$ gilt.

VERTEX COVER

$$(P)\ \text{minimiere} \sum_{u \in V} w_u \cdot x_u \qquad \text{s.d. } x_u + x_v \geqslant 1 \text{ für alle } \{u, v\} \in E$$

$$\text{und } x_u \geqslant 0 \text{ für alle Knoten } u \in V.$$

$$(D)\ \text{maximiere} \sum_{e \in E} y_e, \qquad \text{s.d.} \sum_{x,\ e=\{x,u\} \in E} y_e \leqslant w_u \text{ für alle } u \in V$$

$$\text{und } y_e \geqslant 0 \text{ für alle } e \in E.$$

Erhöhe y_e solange, bis eine duale Ungleichung exakt erfüllt wird.

- Wenn $e = \{u, v\}$, dann taucht y_e nur in den Ungleichungen $\sum_{x,\ e=\{x,u\} \in E} y_e \leqslant w_u$ und $\sum_{x,\ e=\{x,v\} \in E} y_e \leqslant w_v$ auf.
- Wird Gleichheit für die erste Ungleichung erreicht:
  - Setze $x_u = 1$, füge also u zur Knotenüberdeckung hinzu.
  - Ansonsten setze $x_v = 1$, füge also v zur Knotenüberdeckung hinzu.

Ein primal-dualer Algorithmus für VERTEX COVER

(1) Die Eingabe besteht aus dem ungerichteten Graphen $G = (V, E)$ und den Knotengewichten $w_u \geqslant 0$ für $u \in V$.

(2) Beginne mit $V' = \emptyset$ und dem Vektor $y = 0$.

(3) Wiederhole solange, bis V' eine Knotenüberdeckung ist:

(3a) Sei $e = \{u, v\}$ eine nicht von V' überdeckte Kante.
Erhöhe y_e solange, bis eine der Ungleichungen

$$\sum_{x,\ e=\{x,u\}\in E} y_e \leqslant w_u \text{ oder } \sum_{x,\ e=\{x,v\}\in E} y_e \leqslant w_v$$

mit Gleichheit erfüllt wird.

(3b) Wenn Gleichheit für die duale Ungleichung zu u gilt, dann setze $V' = V' \cup \{u\}$ und ansonsten $V' = V' \cup \{v\}$.

Kommentar: Die duale Lösung bestimmt, welcher Endpunkte der nicht überdeckten Kante in die Knotenüberdeckung gelangt.

Der primal-duale Algorithmus ist 2-approximativ

- Sei V' die berechnete Knotenüberdeckung.
  - Für jeden Knoten $u \in V'$ gilt $\sum_{x,\, e=\{x,u\}\in E} y_e = w_u$.
  - und deshalb ist

$$\sum_{u\in V'} w_u = \sum_{u\in V'} \sum_{x,\, e=\{x,u\}\in E} y_e \leqslant \sum_{v\in V} \sum_{x,\, e=\{x,v\}\in E} y_e = 2 \cdot \sum_{e\in E} y_e.$$

  denn y_e wird für jeden Endpunkt von e einmal gezählt.
- $\sum_{e\in E} y_e$ ist der Wert der dualen Lösung.
- Mit dem schwachen Dualitätssatz folgt

$$\sum_{e\in E} y_e \leqslant \sum_{v\in V} w_v x_v^* \leqslant \mathrm{vc}^*(G)$$

  für die optimale primale Lösung x^*.

Also gilt $\sum_{u\in V'} w_u \leqslant 2 \cdot \sum_{e\in E} y_e \leqslant 2 \cdot \mathrm{vc}^*(G)$ und dies war zu zeigen.

Primal-duale und Local Ratio Algorithmen

Im Local Ratio Ansatz für VERTEX COVER:

- Wir beginnen mit einer beliebigen Kante $e = \{u, v\}$,
- nehmen den Endpunkt mit kleinstem Gewicht in unsere Überdeckung auf und
- erniedrigen das Gewicht des anderen Endpunkts um $\min\{w_u, w_v\}$.

Und wenn auch der primal-duale Algorithmus mit Kante $\{u, v\}$ beginnt?

- Die Variable y_e wird auf $\min\{w_u, w_v\}$ gesetzt, und der Endpunkt mit kleinerem Gewicht wird in die Überdeckung aufgenommen.
- Das Hochsetzen von y_e wirkt sich nur in der Ungleichung für den nicht aufgenommen Endpunkt aus:

  Die rechte Seite wird um $\min\{w_u, w_v\}$ erniedrigt.

Und wenn die anderen Kanten verarbeitet werden?

Wir nehmen an, dass der primal-duale Algorithmus und der Local-Ratio Algorithmus Kanten in identischer Reihenfolge wählen.

- Der zweite und alle folgenden Schritte verlaufen für beide Algorithmen analog zum ersten Schritt!
- Die beiden Algorithmen sind äquivalent!

Man kann allgemein zeigen:

Die Paradigmen „primal-duale Algorithmen“ und „Local Ratio Algorithmen“, sind äquivalent, wenn geeignet definiert.

SET COVER

- Mengen $T_1, \ldots, T_m$ mit Gewichten $g_1, \ldots, g_m \geqslant 0$ sind gegeben.
- Wir suchen eine leichteste Überdeckung des Universums $\bigcup_{i=1}^{m} T_i$.

SET COVER für die Ursachenforschung.

- Für die Virenerkennung liegen SEHR VIELE 3-Byte langen Sequenzen vor, die in Viren vorkommen.
  - Wir fassen die (bekannten) Viren als Elemente
  - und jede verdächtige Sequenz als die Menge der Viren auf, in denen die Sequenz vorkommt.
- Ziel: Erstelle eine Software, die aufgrund des Vorkommens verdächtiger Sequenzen lernt, unbekannten Viren vorauszusagen.
  - Das Problem: Die große Anzahl verdächtiger Viren und damit
  - die enorm große Anzahl an Freiheitsgraden für Lernalgorithmen.
- Bestimme eine Überdeckung aller Viren mit möglichst wenigen verdächtigen Sequenzen mit Hilfe von SET COVER.
  - Das abschließende Lernproblem war machbar, da nur einige Hundert verdächtige Sequenzen für die Überdeckung ausreichten.

Ein Approximationsalgorithmus für SET COVER

(1) Gegeben sind die Mengen $T_1, \ldots, T_m \subseteq \{1, \ldots, n\}$ und ihre Gewichte $g_1, \ldots, g_m$.

(2) Setze $U_0 = \emptyset$, $I = \emptyset$ und $i = 0$.

(3) Wiederhole, solange wie $U_i \neq \{1, \ldots, n\}$:

(3a) $| T_k \setminus U_i |$ ist die Anzahl noch nicht überdeckter Elemente von T_k. Bestimme eine Menge T_K mit bestem „Preis-Leistungsverhältnis“

$$\frac{g_K}{|T_K \setminus U_i|} = \min_r \left\{ \frac{g_r}{|T_r \setminus U_i|} \right\}.$$

(3b) Setze $U_{i+1} = U_i \cup T_K$, $I = I \cup \{K\}$, $i = i + 1$.

Primales und duales Programm für SET COVER

SET COVER besitzt das primale „Überdeckungsprogramm“

$$\text{minimiere } \sum_{k=1}^{m} g_k \cdot x_k\text{:} \qquad \sum_{k,\, j \in T_k} x_k \geqslant 1 \text{ für alle } j \in \{1, \ldots, n\}$$
$$\text{und } x_k \geqslant 0 \text{ für } k = 1, \ldots m.$$

und das duale „Packungsprogramm“

$$\text{maximiere } \sum_{j=1}^{n} y_j\text{:} \qquad \sum_{j \in T_k} y_j \leq g_k \text{ für } k = 1, \ldots, m,$$
$$y_j \geq 0 \text{ für } j \in \{1, \ldots, n\}.$$

Überraschenderweise berechnet der Approximationsalgorithmus implizit eine Lösung des dualen Packungsprogramms.

SET COVER und das duale Programm

Sei $H_k = \sum_{r=1}^{k} \frac{1}{r}$ die $k$te harmonische Zahl.

- Element j werde erstmalig in Iteration $i(j)$ durch $T_{k(j)}$ überdeckt.
- Wir weisen Element j die Überdeckungskosten

$$\text{preis}(j) = \frac{g_{k(j)}}{|T_{k(j)} \setminus U_{i(j)}|} \text{ zu.}$$

- **Behauptung**: $y = (\frac{\text{preis}(j)}{H_n} \mid 1 \leqslant j \leqslant n)$ löst das duale Problem.

- Angenommen die Behauptung stimmt.
- Sei $I \subseteq \{1, \ldots, m\}$ die berechnete Überdeckung. Dann ist

$$\sum_{j=1}^{n} \text{preis}(j) = \sum_{j=1}^{n} \frac{g_{k(j)}}{|T_{k(j)} \setminus U_{i(j)}|} = \sum_{i \in I} \sum_{j=1, k(j)=i}^{n} \frac{g_{k(j)}}{|T_{k(j)} \setminus U_{i(j)}|} = \sum_{i \in I} g_i.$$

Wenn preis/H_n das duale Programm löst:

Wir wissen: $\sum_{j=1}^{n} \mathrm{preis}(j) = \sum_{i \in I} g_i$.

- y ist eine Lösung des dualen Programms: Mit der schwachen Dualität folgt

$$\sum_{j=1}^{n} y_j = \frac{1}{H_n} \cdot \sum_{j=1}^{n} \mathrm{preis}(j) \leqslant \mathtt{opt}.$$

- Und als Konsequenz

$$\sum_{i \in I}^{n} g_i = \sum_{j=1}^{n} \mathrm{preis}(j) \leqslant H_n \cdot \mathtt{opt}.$$

Wenn y eine Lösung des dualen Programms ist,
dann ist unser Algorithmus H_n-approximativ.

Zeige $\sum_{j \in T_k} y_j \leqslant g_k$ für jedes k

Zur Erinnerung: $y = (\frac{\text{preis}(j)}{H_n} \mid 1 \leqslant j \leqslant n)$.

- Es sei $T_i = \{e_1, \ldots, e_r\}$, wobei die Elemente von T_i nach dem Zeitpunkt ihrer Überdeckung sortiert sind.
- Vor der Überdeckung von Element e_j sind noch mindestens $r - (j-1) = r - j + 1$ Elemente von T_i zu überdecken.
- Greedy wählt das beste Preis-Leistungsverhältnis. Wenn also Element j zum Zeitpunkt $k(j)$ überdeckt wird, folgt

$$\text{preis}(j) = \frac{g_{k(j)}}{|T_{k(j)} \setminus U_{i(j)}|} \leqslant \frac{g_i}{r - j + 1}.$$

- Also gilt

$$\sum_{j \in T_i} \text{preis}(j) \leqslant g_i \cdot \sum_{j=1}^{r} \frac{1}{r - j + 1} \leqslant g_i \cdot H_n$$

und das war zu zeigen.