RWTH Aachen – Sommersemester 2007

Zusammenfassung

Optimierung B*

Ulrich Loup
Ulrich.Loup@rwth-aachen.de

22. März 2008

Anmerkungen:

- Ich garantiere weder Vollständigkeit noch Korrektheit meiner Angaben.
- Dieses Dokument ist primär als persönliches Nachschlagewerk erstellt worden.
- Verbesserungsvorschläge oder sonstige Bemerkungen nehme ich gerne entgegen.

*Gehalten von Prof. Dr. Eberhard Triesch.

Inhaltsverzeichnis

Literatur

[VolkmannGT] *Graphentheorie*, Lutz Volkmann, Vorlesungsskript, RWTH Aachen 2006.

[LoupGT] *Zusammenfassung Graphentheorie*, Ulrich Loup, private Zusammenfassung, 2005.

[LoupGC] *Zusammenfassung Gitter und Codes*, Ulrich Loup, private Zusammenfassung, 2008.

Übersicht

In Abschnitt 1 werden erforderliche Begriffe definiert und motiviert. In Abschnitt 2 werden die Hauptaussagen zu den entsprechenden Begriffen formuliert, welche dann in Abschnitt 3 für Problemlösungsstrategien verwendet werden.
Intensiv werden Matching-Probleme und Abwandlungen derer sowie Probleme aus der ganzzahligen linearen Optimierung behandelt.

1 Grundlagen

1.1 Graphen

Die Begriffe der Graphentheorie sind detailliert aus [VolkmannGT] oder [LoupGT] entnehmbar. In der folgenden Bemerkung werden nur die speziell hier verwendeten Notationen zusammengefasst.

Bemerkung 1.

- *Graphen $G = (V, E)$ sind hier endlich,* ***schlicht***[1] *und* ***ungerichtet***[2]*. Die Kantenschreibweise $e = \{x, y\} \in E$ wird durch xy abgekürzt.*
- *Der Graph $G = (V, E)$ induziert die* ***Inzidenzmatrix*** *$B(G) = (b_{ij})$ mit $b_{ij} = (v_i \in e_j)$ sowie die* ***Adjazenzmatrix*** *$A(G) = (a_{ij})$ mit $a_{ij} = (v_i, v_j \in E)$, wobei $v_i, v_j \in V$ und $e_j \in E$.*
- *Ein zusammenhängender Graph ohne Kreise heißt* ***Baum****, ein kreisfreier Graph*[3] *heißt* ***Wald****.*

Erinnerung 1. *Sei $G = (V, E)$ ein Graph.*

- *$(V', E') = H \subseteq G$ heißt* ***Teilgraph*** *oder* ***Untergraph*** *in G, falls $V' \subseteq V$ und $E' \subseteq E$.*[4]
- *G heißt* ***bipartit****, falls $V = U \uplus W$ eine Partition von V ist, so dass für alle $xy \in E$ gilt: $x \in U$ und $y \in W$ oder $x \in W$ und $y \in U$.*[5]
- *Eine* ***Kantenfolge*** *ist eine Folge $x_0 e_1 x_1 e_1 \dots x_r$ mit $x_i \in V$ und $e_i \in E$ sowie $x_{i-1} x_i = e_i$. r heißt die* ***Länge der Kantenfolge****.*
- *Ein* ***Kantenzug*** *ist eine Kantenfolge $x_0 e_1 x_1 e_1 \dots x_r$ mit $e_i \notin \{e_j \mid 1 \leq j \neq i \leq r\}$ für alle i.*

[1] Das heißt, sie haben keine Schlingen und Mehrfachkanten.
[2] Das heißt, $E \subseteq \binom{V}{2} := \{M \subseteq V \mid |M| = 2\}$.
[3] nicht notwendig zusammenhängend
[4] Die Begriffe $G[U]$ für $U \subseteq V$ oder $U \subseteq E$ bezeichnen jeweils die von U induzierten Untergraphen.
[5] Siehe Definition und Bemerkung 19

- *Ein* ***Weg*** *ist ein Kantenzug* $x_0e_1x_1e_1\dots x_r$ *mit* $x_i \notin \{x_j \mid 0 \leq j \neq i \leq r\}$ *für alle* i.
- *Ein* ***Kreis*** *ist ein Kantenzug* $x_0e_1x_1e_1\dots x_r$ *mit* $x_i \notin \{x_j \mid 1 \leq j \neq i \leq r-1\}$ *für alle* i *und* $x_0 = x_r$.
- *Ein Kantenzug* $x_0e_1x_1e_1\dots x_r$ *heißt* **Eulersch**, *falls* $r = |E|$.
- *Ein Kreis der Länge* $|V|$ *und ein Graph, der einen solchen Kreis besitzt heißen jeweils* **Hamiltonsch**.
- *Für* $U \subseteq V$ *heißt* $\Gamma(U) := \{x \in V \mid ux \in E \text{ und } u \in U\}$ *die* ***Menge der Nachbarn von*** U.

Die Begriffe **minimum** und **maximum** beziehen sich stets auf Minimalität und Maximalität der Kardinalität, **minimal**,**maximal** auf die entsprechenden Begriffe bezüglich Inklusion, das bedeutet, eine minimale Menge mit einer bestimmten Eigenschaft ist das kleinste Element einer Inklusionskette mit dieser Eigenschaft. Eine minimum Menge mit einer bestimmten Eigenschaft ist die kleinste Menge unter allen Mengen mit dieser Eigenschaft.

Definition 1. *Seien* $G = (V, E)$ *ein Graph und* $H \subseteq G$ *ein Teilgraph.*

- H *heißt* ***aufspannend***, *falls* $V' = V$.
- *Sind die Kanten in* G *vermöge* $c : E \to \mathbb{R}$ *bewertet, so heißt ein aufspannender Baum mit Minimalen Gesamtkosten ein* ***minimum spanning tree (MST)***.

Definition 2 (Matchings und vertex cover). *Sei* $G = (V, E)$ *ein Graph.*

- *Eine Kantenmenge* $M \subseteq E$ *heißt* ***matching in*** G, *falls* $e \neq f$ *oder* $e \cap f = \emptyset$ *für alle* $e, f \in M$.[6]

  $\nu(G) := \max\{|M| \mid M \text{ matching in } G\}$.
- *Ein maximum matching* M *in* G *heißt* ***perfekt***, *falls* $\bigcup M = V$ *ist, wobei* $\bigcup M = \{x, y \in V \mid xy \in M\}$.[7]
- *Eine (Zusammenhangs)komponente* H *in* G *heißt* ***faktorkritisch***, *falls* $H - x$ *ein perfektes matching besitzt für alle* $x \in V(H)$.
- *Eine Knotenmenge* $U \subseteq V$ *heißt* ***vertex cover in*** G, *falls* $e \cap U \neq \emptyset$ *für alle* $e \in E$.

  $\tau(G) := \min\{|U| \mid U \text{ vertex cover von } G\}$.

  G *heißt* τ***-kritisch***, *falls* $\tau(G - e) = \tau(G) - 1$ *für alle* $e \in E$.

[6] Merke: e, f sind 2-Teilmengen der Knotenmenge.

[7] Ein perfektes matching ist ein 1-Faktor, also ein Teilgraph in G, der alle Ecken in G enthält, welche alle den Grad 1 haben.

Definition und Bemerkung 3 (Graphparameter bezüglich Zusammenhang).
Sei $G = (V, E)$ ein Graph.

- $q(G) = |\{C \subseteq G \mid C \text{ Zusammenhangskomponente mit } |V(C)| \text{ ungerade}\}|$
- $\kappa(G)$ *bezeichnet die Anzahl der Zusammenhangskomponenten von* G.
- *Eine Kante* $e \in E$ *mit* $\kappa(G - e) > \kappa(G)$ *heißt* ***Brücke***.
- *Eine Ecke* $x \in V$ *mit* $\kappa(G - x) > \kappa(G)$ *heißt* ***trennende Ecke***.
- *Sind* G *bitpartiter Graph und* $V = U \uplus W$ *sowie* M *ein maximum matching und* Z *ein minimum vertex cover in* G, *so gilt:*
  1. $x \in \bigcup M$ *für jede trennende Ecke* $x \in V$,
  2. $e \cap Z \neq \emptyset$ *für jede Brücke* $e \in E$.
- *Jeder maximale Weg in* G *besitzt jede trennende Ecke und jede Brücke in* G.

1.1.1 Netzwerke

Netzwerke sind gerichtete Graphen mit Bogenbewertung [8] und den ausgezeichneten Ecken „Quelle“ und „Senke“.

Definition 4.

- $D = (V, A)$ *heißt* ***gerichteter Graph***, *falls* $A \subseteq V \times V$. A *ist die Menge der* ***Bögen***[9] *von* D.
- (D, s, t, c) *heißt* ***Netzwerk***, *falls* $D = (V, A)$ *ein gerichteter Graph ist und* $c : A \to \mathbb{R}_{\geq 0}$ *Kapazitätsfunktion und es gilt*
  1. $s, t \in V$,
  2. $(s, t) \in A$ *mit* $\sum_{a \in A \setminus \{(t,s)\}} c(a) < c((t, s))$.

  s *heißt* ***Quelle*** *und* t ***Senke*** *von* D.
- *Sei* (D, s, t, c) *mit* $D = (V, A)$ *ein Netzwerk.*
  - $x : A \to \mathbb{R}_{\geq 0}$ *beziehungsweise* $x \in \mathbb{R}^{|A|}$ *heißt* ***Fluss***, *falls* $\sum_i \sum_j b_{ij} x_j = Bx = 0$. x *heißt* ***zulässig***, *falls* $0 \leq x(a) \leq c(a)$ *für alle* $a \in A$.

    $\mathrm{val}(x) := x((t, s))$ *heißt der* ***Wert*** *von* x.
  - $C \subseteq V$ *heißt* ***Schnitt***, *falls* $s \in C$ *und* $t \notin C$.

    $\mathrm{cap}(C) := \sum_{(u,v) \in (C \times (V \setminus C)) \cap A} c((u, v))$ *heißt die* ***Kapazität*** *von* C.

[8] Kapazitäten $\hat{=}$ obere Schranke für den Fluss
[9] Englisch: arks

– *Sei x ein Fluss. Ein gerichteter Weg $v_0 a_1 v_a \dots a_k v_k$*[10] *heißt* ***vergrößernder s-t-Weg****, falls $v_0 = s$, $v_k = t$ und $c(a) - x(a) > 0$*[11] *für jede Vorwärtskante $a \in A$ sowie $x(a) > 0$ für jede Rückwärtskante $a \in A$.*

1.2 Matrizen

Definition und Bemerkung 5.

- *Eine Matrix $(a_{ij}) = A \in \{0,1\}^{n \times n}$ heißt* ***Permutationsmatrix****, falls $\sum_j a_{ij} = 1$ für alle i, $\sum_i a_{ij} = 1$ für alle j.*

  Alternativ: *$P_\pi \in \Omega_\pi$ Permutationsmatrix, falls $(P_\pi)_{ij} = \begin{cases} 1 & \pi(i) = j \\ 0 & sonst \end{cases}$ für $\pi \in S_n$.*

  Eine Permutationsmatrix $A \in \Omega_n$ hat genau $p(A) = n$ postitive Einträge.

- *Eine Matrix $(a_{ij}) = A \in \mathbb{R}^{n \times n}$ heißt* ***doppelstochastisch****, falls $a_{ij} \geq 0$ für alle i, j und $\sum_j a_{ij} = 1$ für alle i, $\sum_i a_{ij} = 1$ für alle j.*[12] *Ω_n ist die Menge der doppelstochastischen Matrizen.*

- *Eine* ***Konvexkombination*** *Für Matrizen $A_1, \dots, A_k$ ist $\sum_{i=1}^k b_i A_i$ eine* ***Konvexkombination****, falls $b_i \geq 0$ und $\sum_i b_i = 1$.*

- *Für eine Menge $X \subseteq \mathbb{R}^n$ heißt die Menge aller Konvexkombinationen*

$$\mathcal{C}(X) := \{\sum_{i=1}^k a_i x_i \mid k \in \mathbb{N}, x_i \in X, a_i \geq 0, \sum_{i=1}^k a_i = 1\}$$

  die ***konvexe Hülle von*** *X.*

Der folgende Begriff ist ähnlich wie die Determinante einer Matrix definiert, jedoch ohne das alternierende Vorzeichen wie bei der Entwicklung einer Determinante.

Definition 6. *Sei $(a_{ij}) = A$ eine $n \times n$-Matrix.*

- $\operatorname{per}(A) := \sum_{\pi \in S_n} \prod_{i=1}^n a_{i\pi(i)}$ *heißt die* ***Permanente von*** *A.*[13]

- *Die* ***Permanente*** *einer Sudoko-Matrix ist konstant $9! \cdot 362880$.*

Im Umgang mit Matrizen werden oft eine oder mehrere Zeilen und Spalten gelöscht, was die folgenden Notationen vereinfacht darstellen lassen.

Definition 7. *Sei A eine Matrix.*

- *$A(I|J)$: Matrix A ohne i-ten Zeilen und j-ten Spalten für $i \in I$, $j \in J$.*

[10] das bedeutet,
[11] Restkapazität
[12] Das heißt, alle Zeilensummen und alle Spaltensummen sind 1.
[13] Zum Vergleich: Nach der Leibnitz-Formel ist $\det(A) = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i=1}^n a_{i\pi(i)}$.

- *$A(i|j)$: Matrix A ohne i-te Zeile und j-te Spalte.*
- *$A[I|J]$: Matrix A genau mit i-ten Zeilen und j-ten Spalten für $i \in I$, $j \in J$.*[14]
- ***Spaltenschreibweise:*** *$A = (a_1, \ldots, a_n)$ oder $B = (A, a1, \ldots, a_m)$.*

Um die Matrizen mit positiven oder Null-Permanenten zu charakterisieren, werden folgende Begriffe benötigt.

Definition 8. *Sei $A \in \Omega_n$. A heißt genau dann **zerlegbar**, wenn A durch Zeilen- und Spaltenpermutationen auf die Form $A = \begin{pmatrix} B & 0 \\ C & D \end{pmatrix}$ gebracht werden kann, wobei B, C quadratische Matrizen sind. Ansonsten heißt A **unzerlegbar**.*

Die folgende Definition beschreibt das minimale Element von Ω_n bezüglich der durch $\leq$ angeordneten Permanenten.

Definition 9. *Die Matrix $A \in \Omega_n$ heißt **minimierend**, falls* $\operatorname{per}(A) \leq \min_{B \in \Omega_n} \operatorname{per}(B)$.

1.3 Lineare Optimierung

Das grundlegende Problem in der linearen Optimierung ist ein Minimierungs- oder Maximierungsproblem mit linearen Nebenbedingungen (oder: Restriktionen), im Folgenden als Maximierungsproblem formuliert.

Problem 1 (LP).

Gegeben:	*Kostenvektor $c \in \mathbb{R}^n$ und $b \in \mathbb{R}^n$, $(a_{ij}) = A \in \mathbb{R}^{m \times n}$.*
Gesucht:	*Ein $x \in \mathbb{R}^n$, so dass die **Zielfunktion** $c^T x$ maximal wird unter den*
	Nebenbedingungen: $Ax \leq b$
	$x \geq 0$

Dies kann auch kürzer in folgender Form notiert werden:

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

Insbesondere wird durch $Ax \leq b$ ein lineares Ungleichungssystem mit m Ungleichungen der Form $a_i x = b_i$ aufgestellt, wobei a_i die Zeilen von A sind. Im Kontext der LP sind noch einige Begriffe geläufig.

Definition und Bemerkung 10.

- Ungleichungen:

  scharf: $a_i x < b_i$

[14] Das heißt, $A[I|J] = A(I^C|J^C)$ für die Komplemente I^C, J^C von I respektive J.

aktiv: $a_i x = b_i$

- Lösungen:

  unbeschränkt: $Ax \geq b$ *gilt für* unendlich viele $x \in \mathbb{R}$.

  unzulässig: $Ax \leq b$ *gilt für* kein $x \in \mathbb{R}^n$.

- Charakterisierung des Lösungsraums:
  - $\{x \in \mathbb{R}^n \mid a_i x \leq b_i\}$ *ist ein Halbraum in* $\mathbb{R}^n$.
  - $\{x \in \mathbb{R}^n \mid a_i x = b_i\}$ *ist eine Hyperebene in* $\mathbb{R}^n$.
  - $\bigcap_{i=1}^m \{x \in \mathbb{R}^n \mid a_i x \leq b_i\}$ *ist der Lösungsraum für die Zielfunktion und ein* ***Polyeder*** *in* $\mathbb{R}^n$, *das* ***Lösungspolyeder*** *des Problems.*

Minimierungs- und Maximierungsprobleme sind äquivalent in folgendem Sinne.

Definition und Bemerkung 11.

- *Gegeben seien die folgenden LPs:*

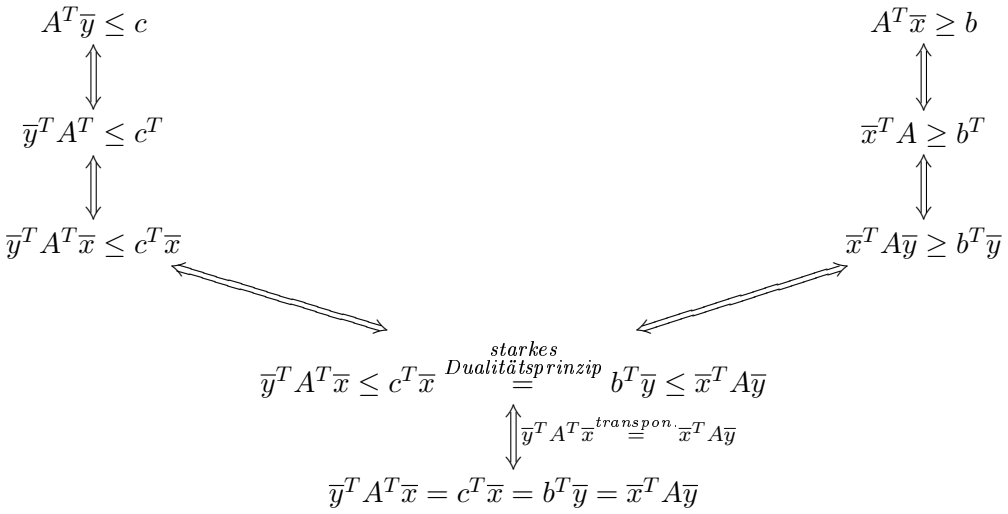
$$\begin{array}{lrcl} (P): & \max & c^T x & \\ & Ax & \leq & b \\ & x & \geq & 0 \end{array} \qquad \begin{array}{lrcl} (D): & \min & b^T y & \\ & A^T y & \geq & c \\ & y & \geq & 0 \end{array}$$

  (P) *heißt das* ***primale*** *und* (D) *das* ***zu*** (P) ***duale*** *Problem.*

- (P) *unbeschränkt* $\Longrightarrow$ (D) *unzulässig.*
- Schwaches Dualitätsprinzip: *Sind* $\overline{x}, \overline{y} \in \mathbb{R}^n$ *zulässige Lösungen von* (P) *respektive* (D), *so ist* $c^T \overline{x} \leq b^T \overline{y}$.
- Starkes Dualitätsprinzip: *Sind* $\overline{x}, \overline{y} \in \mathbb{R}^n$ *optimale Lösungen von* (P) *respektive* (D), *so ist* $c^T \overline{x} = b^T \overline{y}$.
- *Sei* P *ein Problem. Das duale des zu* P *dualen Problems ist* P *selbst.*

Aus der Dualität kann eine äquivalente Bedingung, der komplementäre Schlupf, für die Lösbarkeit der Probleme hergeleitet werden.

Bemerkung 2 (Complementary Slackness).
Seien $\overline{x}, \overline{y} \in \mathbb{R}^n$ Lösungen für das primale respektive das duale LP. Es gilt:



Also gilt $0 = c^T\overline{x} - \overline{y}^T A\overline{x} = \underbrace{(c^T - \underbrace{\overline{y}^T A}_{\geq c})}_{\geq 0} \underbrace{\overline{x}}_{\geq 0}$, *womit die positiven Komponenten der Lösung die aktiven Ungleichungen induzieren.*

Eine besonders schwer zu lösende Klasse der linearen Probleme wird in Abschnitt 2.3 aufgeführt.
Der Lösungsraum M eines LPs kann als Schnitt aller durch die Nebenbedingungen aufgespannten Halbräume gesehen werden, ist also geometrisch betrachtet ein Polyeder. Mit dieser Anschauung kann ein Verfahren zur Lösung des LPs angegeben werden, da es stets einen optimalen Eckpunkt in M gibt.

1. Starte mit $i := 0$ und bei einem beliebigen Eckpunkt v_i.
2. Wähle den nächsten Eckpunkt v_{i+1} entlang der Kante des Polyeders, in deren Richtung sich der Zielfunktionswert verbessert.
3. Stoppe, falls keine verbessernde Kante mehr existiert. Der aktuelle Eckpunkt liefert einen optimalen Zielfunktionswert.

Ein bekanntes Beispiel für Lineare Optimierungsprobleme ist neben dem Flußproblem das *Rucksackproblem*, wovon im folgenden eine Variante vorgestellt wird, in welcher ein Rucksack mit n Gegenständen gefüllt werden soll, so dass möglichst viel bei möglichst wenig Gesamtgewicht im Rucksack ist.

Problem 2 (Rucksackproblem).

Gegeben:	*Gewichte $g_i \in \mathbb{R}$, Platzbedarfe $p_i \in \mathbb{R}$ für jeden Gegenstand $1 \leq i \leq n$, Platz des Rucksacks $P \in \mathbb{R}$.*
Gesucht:	*Zahlen $x_i \in \mathbb{R}$ für $1 \leq i \leq n$, die angeben, wie oft der i-te Gegenstand eingepackt werden soll.*

Beispiel 1. *Die Zielfunktion, welche zu minimieren ist, ergibt sich zu der Summe aller Gewichte*

$$f(x) = \sum_{i=1}^{n} g_i x_i.$$

Die Nebenbedingung ist die Forderung, dass alle Gegenstände zusammen nicht den Gesamtplatz überschreiten dürfen:

$$\sum_{i=1}^{n} p_i x_i \leq P$$

1.4 Komplexitätstheorie

Der hier verfolgte Ansatz zur Betrachtung von Komplexitätsklassen ist die Verwendung von Turing-Maschinen. In folgender Bemerkung werden die wichtigsten Begriffe auf intuitiver Ebene festgelegt.

Bemerkung 3.

(deterministische) Turing-Maschine:

Tupel (Alphabet A, Zustandsraum Z, Übergangsfunktion δ, Startzustand z_0, Endzustandsmenge F).

Konfiguration: *Wort aus A^*ZAA^*.*

(endliche) Rechnung: *($Konfiguration_1$,..., $Konfiguration_T$), T ist die Länge der Rechnung. Die Rechnung heißt **kanonisch**, falls $Konfiguration_1 = z_0 w$ für ein Eingabewort $w \in A^*$.*

Ist der die Länge der Berechnung polynomiell in der Länge der Eingabe bezüglich eines Polynoms $t(n)$, so heißt die zugehörige Turing*-Maschine $t(n)$-*zeitbeschränkt.

Bandverbrauch: *maximale Länge aller Bandinschriften während einer Berechnung:*

$$\max\{|Konfiguration_i| \mid i \in \{1, \ldots, T\}\}.$$

Ist der Bandverbrauch polynomiell in der Länge der Eingabe bezüglich eines Polynoms $t(n)$, so heißt die zugehörige Turing*-Maschine $t(n)$-*bandbeschränkt.

Akzeptor: Turing*-Maschine mit Zuständen z_- und z_+, die eindeutig einen Miserfolg oder Erfolg kennzeichnen (Entscheidungs-*Turing*-Maschine).*

nichtdeterministische TURING-Maschine: *eine Konfiguration kann mehrere Nachfolgekonfigurationen haben, genauer ist* $\delta \subseteq Z \times A \times (Z \times A \times \{R, L, N\}$.

Bemerkung 4.
Eine $t(n)$ *zeitbeschränkte* TURING*-Maschine ist* $t(n) + n$*-bandbeschränkt.*

Beweis: $t(n)$: Je Berechnungsschritt wird maximal ein Zeichen hinzugefügt.
n: Länge des Eingabewortes. □

Jedes Problem kann in ein äquivalentes Entscheidungsproblem umformuliert werden, weswegen die Problemklassen auch als Sprachklassen definiert werden können.
Die beiden wichtigsten Problemklassen sind:

$\mathcal{P}$: Alle durch einen deterministischen, $t(n)$-zeitbeschränkten Akzeptor entscheidbaren Sprachen.

$\mathcal{NP}$: Alle durch einen nichtdeterministischen, $t(n)$-zeitbeschränkten Akzeptor entscheidbaren Sprachen.

Offensichtlich ist $\mathcal{P} \subseteq \mathcal{NP}$. Offen: $\mathcal{P} \neq \mathcal{NP}$?

Definition 12.
- *Seien* L_1, L_2 *Sprachen.* L_1 *ist auf* L_2 ***reduzierbar*** *(*$L_1 \propto L_2$*), falls eine polynomiell berechenbare Funktion* f *zwischen beiden Alphabeten besteht, so dass* $l \in L_1 \iff f(l) \in L_2$.[15]
- *Eine Sprache* L *heißt* $\mathcal{NP}$***-vollständig***, *wenn* $L \in \mathcal{NP}$ *und* L *ist* $\mathcal{NP}$*-schwer, das bedeutet,* $L' \propto L$ *für alle* $L' \in \mathcal{NP}$.

Da die Zugehörigkeit zu $\mathcal{NP}$ für viele Probleme leicht nachzuweisen ist, genügt es die $\mathcal{NP}$-Schwere nachzuweisen. Dies ist aber erreichbar durch die Reduktion eines bereits als $\mathcal{NP}$-vollständig charakterisierten Problems. Die Schwierigkeit liegt nun darin, ein passendes Problem dieser Art zu finden.

1.5 Approximation

Viele NP-vollständige Probleme können durch einige Zusatzbedingungnen an die Eingabe effizienter gelöst werden. In einigen Fällen ist auch eine leichte Abweichung des Ergebnisses von dem exakten Optimalwert vertretbar, was zu dem Konzept der Approximation führt: **Approximationsalgorithmen** liefern ein um ein kleines ϵ abweichendes Ergebnis vom Optimalwert.

[15] Das bedeutet, L_2 ist leichter als L_1 oder gleich schwer.

Betracht hier exemplarisch eine Instanz I eines Minimierungsproblems

$$\begin{aligned} \min \quad & c^T x \\ Ax &\leq b \\ x &\geq 0. \end{aligned}$$

Zur Bestimmung der Güte eines Approximationsalgorithmus werden folgende Begriffe benötigt.

Definition 13.
- $\mathrm{Opt}(I)$ *bezeichnet den optimalen Wert der Zielfunktion.*
- *Ein Algorithmus A liefert die Lösung $A(I)$ zu I.*
- *Der Quotient $R(I) := \frac{A(I)}{\mathrm{Opt}I}$ ist **Gütemaß** für einen Algorithmus A.*[16]

Ein Approximationsalgorithmus A liefert Werte um $\mathrm{Opt}(I)$ herum, das bedeutet, $R(I) \pm \varepsilon' = 1$. Etwas anders formuliert ist die Güte eines Approximationsalgorithmus

$$R(I) = 1 + \varepsilon$$

für ein möglichst kleines $|\varepsilon|$.
Ein Beispiel für ein durch Approximation lösbares *Travelling Salesman Problem* mittels Einschränkung der Eingabe wird in Abschnitt 3.5 behandelt.

2 Ergebnisse

Hier werden aufbauend auf die in Abschnitt 1 eingeführten Begriffe einige Aussagen bewiesen, welche in Abschnitt 3 Anwendung finden.

2.1 Graphen

Lemma 1 (Handschlagslemma). *Für jeden Graph $G = (V, E)$ ist $\sum_{x \in V} d(x) = 2|E|$.*

Beweis: Jede Kante leistet Beitrag 2 zu der Summe der Eckengrade, jeder Eckengrad geht mit dem Endpunkt einer Kante einher. □

Das Handschlagslemma kann auch als Gleichung bezüglich der Inzidenzmatrix von G aufgefasst werden, da der Eckengrad gerade die Anzahl der mit der Ecke inzidierenden Kanten, also die Zeilensumme der entsprechenden Zeile der Inzidenzmatrix ist, und die Kanten jeweils immer genau zwei 1en pro Spalte liefern, also pro Kante die Spaltensumme (= 2) der entsprechenden Spalte der Inzidenzmatrix.

[16] Im Falle eines Maximierungsproblems: $\frac{\mathrm{Opt}(I)}{A(I)}$.

Folgerung 1. *Wegen* $2 \mid \sum_{x \in V} d(x) = \sum_{\substack{u \in V \\ d(u) ungerade}} d(u) + \sum_{\substack{v \in V \\ d(v) gerade}} d(v)$ *ist die Anzahl der Ecken ungeraden Grades gerade.*

Eine besondere Klasse von Graphen sind Wälder und insbesondere Bäume. In diesen Graphen ist *zyklomatische Zahl*[17] 0, was in folgender Bemerkung anders foruliert wird.

Bemerkung 5.
In einem Wald G ist die Summe der Anzahl der Zusammenhangskomponenten ($\kappa(G)$) und der Anzahl der Kanten $|E|$ genau $|V|$. Ein Baum hat also genau $|V| - 1$ Kanten.

Satz 1 (Bäume und Wälder).
Sei $G = (V, E)$ ein Graph mit $m = |E|$, $n = |V|$. Folgende Aussagen sind äquivalent.

- *G ist Baum.*
- *G ist zusammenhängend und für jede neue Kante $e \in \binom{V}{2}$ ist $G - e$ ist nicht zusammenhängend, das heißt, G ist **minimal zusammenhängend**.*
- *G ist zusammenhängend, kreisfrei und für jede neue Kante $e \in \binom{V}{2}$ hat $G + e$ einen Kreis, das heißt, G ist **maximaler Wald**.*
- *Sind $x \neq y \in V$, so existiert genau ein x-y-Weg.*
- *G ist zusammenhängend und $m = n - 1$.*
- *G ist Wald und $m = n - 1$.*

Der KRUSKAL-Algorithmus berechnet einen MST zu einem gegebenem Graphen. Dieser ist im Allgemeinen nicht eindeutig. Eine obere Schranke für die Anzahl der aufspannenden Bäume eines Graphen liefert der folgende Satz.

Satz 2 (CAYLEY, 1889).
Ist $G = (V, E)$ ein Graph mit $n = |V|$, so gibt es genau n^{n-2} aufspannende Bäume in G.

Beweis: Sei $V = \{1, \ldots, n\}$.

1. PRÜFER-*Code:* Kodiere einen Spannbaum T von $G = (V, E)$ eindeutig durch eine Folge $(v_1, \ldots, v_{n-2})$ von Knoten, wobei v_1 Nachbar des kleinsten Blattes b_1 von T, v_2 Nachbar des kleinsten Blattes von $T - b_1$, ... bezüglich der Ordnung auf $V = \{1, \ldots, n\}$ ist. Es gibt n^{n-2} Tupel der Form $(v_1, \ldots, v_{n-2})$.

2. *Verallgemeinerte Rekursion:* Für $A \subseteq V$ mit $|A| = k$ definiere

$$\mathcal{T}_{n,A} := \{F = (V, E') \mid F \text{ ist ein Wald, dessen Komponenten je genau ein Element aus } A \text{ enthalten}\}.$$

□

[17] Vergleiche [VolkmannGT], Seite 14

2.1.1 Bipartite matchings

In bipartiten Graphen ist das Auffinden eines minimalen vertex covers genauso aufwendig wie das Bestimmen eines maximum matchings. In nicht bipartiten Graphen ist die Suche nach einem vertex cover NP-schwer, obwohl es mit dem EDMONDS-Algorithmus ein effizientes Lösungsverfahren für das allgemeine matching-Problem gibt.
Der Satz von KÖNIG kann mittels der Suche eines τ-kritischen Teilgraphen eines Graphen bewiesen werden. Folgendes Teilergebnis beschreibt das Ergebnis dieser Suche.

Lemma 2. *Sei $G = (V, E)$ ein bipartiter Graph.*
Ist G τ-kritisch, so gilt $\nu(G) = \tau(G)$.

Beweis: Angenommen $\nu(G) \neq \tau(G)$. Es gibt zwei Fälle. Sei M maximum matching in G.

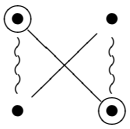
1. $\nu(G) > \tau(G)$: $G[M]$ besitzt ein minimum vertex cover der Größe $\nu(G[M]) = \nu(G)$ ($G[M]$ hat $\nu(G)$ Komponenten.). Da $G[M] \subseteq G$ gilt zum Widerspruch

$$\nu(G) = \tau(G[M]) \leq \tau(G).$$

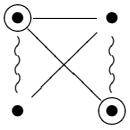
2. $\nu(G) < \tau(G)$: Sei $uv = e \in E \setminus M$. Also $\tau(G - e) = \tau(G) - 1$, das Entfernen von e zerstört auch einen u (o.B.d.A.) Punkt des vertex covers. e induziert eine Hantel. Daher muss $e \in M$ gewesen sein, Widerspruch.

□

Die Umkehrung der Folgerung sowie der Aussage des folgenden Satzes ist im Allgemeinen nicht gültig. Betrachte beispielsweise den folgenden bipartiten Graphen:



Die umkreisten Knoten bilden ein minimum vertex cover und die geschlängelten Kanten ein maximum matching, also ist $\nu = \tau = 2$. Eine simple Modifikation des Graphen ändert die Parameter ν und τ nicht, resultiert aber in einem nicht mehr bipartiten Graphen:



Satz 3 (Dualität: matchings und vertex cover, KÖNIG).
Ist G biparteter Graph, so gilt $\nu(G) = \tau(G)$.

Beweis:

Nicht konstruktiv: (Lovász) Erhalte einen τ-kritischen Teilgraphen $H = (V, F)$ von G durch Wegnehmen von Kanten so lange, wie $\tau(G)$ nicht abnimmt, das heißt, solange alle minimalen vertex cover nicht kleiner werden.[18]

Zeige: F ist ein maximum matching in G mit Kardinalität $\tau(G)$.

- Matching: Angenommen F wäre kein matching, dann gibt es einen Knoten $v \in V$, der mit zwei Kanten $vx, vy \in F$ inzidiert.
- Maximalität:
- Kardinalität: Da H τ-kritisch ist, gilt $|F| = \tau(G)$. Sonst könnte noch eine Kante entfernt werden, ohne dass $\tau(H)$ abnimmt.

Konstruktiv: Die ungarische Methode berechnet zu jedem bipartiten Graphen G ein maximum matching M und ein vertex cover C. Nach Definition des Algorithmus ist $C \setminus (\underbrace{(\bigcup M \cap W)}_{\text{von } M \text{ überdeckte}} \cup (\bigcup (E \setminus M) \cap U) = \emptyset$.

□

Sind die beiden Partitionsmengen eines bipartiten Graphen gleich groß, so ergibt sich ein Spezialfall des matching Problems.

Bemerkung 6.
Sei $G = (U \uplus W, E)$ ein zusammenhängender bipartiter Graph mit $|U| = |W|$.

1. *Ein matching M in G ist genau dann perfekt, wenn $|M| = |U| = |W|$ ist.*
2. *U oder W sind jeweils ein minimum vertex cover in G.*

Beweis:

1. Ist M perfekt, so muss jeder Punkt von U und jeder Knoten von W mit einer Kante von M inzidieren. Da M ein matching ist, sind all diese Kanten verschieden, woraus die Behauptung folgt.

   Umgekehrt sei $|M| = |U| = |W|$. Angenommen M wäre nicht perfekt und $u \in U$ würde daher nicht abgedeckt. Dann existiert eine Kante $u'w \in M$ mit $u' \neq u$. Es gibt aber ebenfalls eine Kante $u'w' \in M$ mit $w' \neq w$. Dann ist M aber kein matching, Widerspruch.

2. Angenommen U (analog für W) wäre kein minimum vertex cover (U oder W sind triviale vertex cover).

□

[18] praktisch schwer durchführbar

Satz 4 (Hall, 1930).
Sei $G = (U \uplus W, E)$ bipartiter Graph. Genau dann existiert ein matching M mit $|M| = U$, wenn für alle $X \subseteq U$ die Hall-Bedingung *gilt:*

$$|\Gamma(X)| \geq |X|$$

Beweis:

Intuitiv: Das geforderte matching M soll alle Punkte von U überdecken. Gäbe es eine Menge von Punkten in U, mit weniger Nachbarn (in W) als die Menge selbst Punkte besitzt, so müssen zwei Kanten existieren, die sich in einem Punkt von $\Gamma(U) \cap W$ treffen.

Umgekehrt bedeutet die Existenz eines U-überdeckenden matchings, dass $\Gamma(X) \cap W$ für jedes X mehr Punkte als X enthält.

Formal: „$\Rightarrow$“: Wie oben beschrieben.

„$\Leftarrow$“: Zum Widerspruchsbeweis wird angenommen, dass jedes matching nicht ganz U überdeckt, also $\nu(G) < |U|$. Dann kann die Dualität zwischen matching und vertex cover (Satz 3) benutzt werden, das heißt, es gibt in dem Fall auch ein vertex cover $Z \subseteq V$ mit $|Z| < |U|$. Mit $X := U \setminus Z$ kann schließlich der Widerspruch hergestellt werden.

□

Der Satz von Hall kann in folgender Bemerkung auf Matrizen angewandt werden.

Bemerkung 7 (Satz von Birkhoff - von Neumann).
Jede doppelstochastische Matrix $A \in \Omega_n$ ist eine Konvexkombination von Permutationsmatrizen P_π für $\pi \in S_n$.

Beweis: Induktion nach der Anzahl $p(A)$ der positiven Elemente von A.

I.A.: $p(A) = n$, nach Definition und Bemerkung 5 ist A also Permutationsmatrix.

I.S.: Sei nun $p(A) > n$ für $A = (a_{ij})$.

Definiere bipartiten Graphen $G = (U \uplus W, E)$ zu A mit $U := \{u_1, \ldots, u_n\}$, $W := \{w_1, \ldots, w_n\}$ und E derart, dass $u_i w_j \in E \iff a_{ij} > 0$.

Behauptung: Es gibt ein maximum matching M in G ($\nu(G) = n$).

Beweis: Überprüfen der Hall-Bedingung für G. Intuitiv bildet schon eine Permutationsmatrix genau ein maximum matching, also insbesondere eine doppelstochastische Matrix.

Das gefundene matching M ist äquivalent zu einer der Permutationsmatrizen P_i aus der zu beweisenden Konvexkombination $A = \sum_{i=1}^{k} \alpha_i P_i$. Wähle α_i als den Wert des

minimalen positiven Elements aus A, welcher einer matching-Kante entspricht und suche in der neuen Matrix $A-\alpha_i P_i$ wieder ein maximum matching wie oben, bis die Nullmatrix übrig bleibt.

□

2.1.2 Nicht bipartite matchings

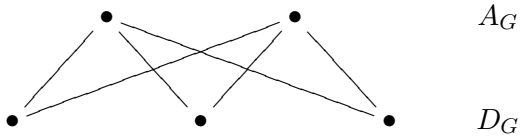
Wichtige Grundlage für das Finden von matchings in nicht bipartiten Graphen ist der folgende Satz über die Struktur von allgemeinen Graphen. Mit diesem Werkzeug können auch schon vorher bekannte Ergebnisse der Graphentheorie bequem gezeigt werden. Der Satz bezieht sich auf folgende Partition von V:

Definition und Bemerkung 14. *Sei $G = (V, E)$ ein Graph.*

- *Setze*

$$\begin{aligned} D_G &:= \{x \in V \mid \text{es ex. ein maximum matching } M \text{ mit } x \notin M\}, \\ A_G &:= \Gamma(D_G) \setminus D_G, \\ C_G &:= V \setminus (D_G \cup A_G). \end{aligned}$$

1. *A_G ist die Menge aller Punkte in G, die von jedem maximum matching überdeckt werden.*
2. *A_G muss nicht notwendiger Weise trennende Ecken enthalten und C_G kann leer sein:*



Beweis:

1. Sei M ein maximum matching von G und $x \in A_G$. Würde x nicht von M überdeckt, so wäre M ein Zeuge für $x \in D_G$.
2. Das Entfernen einer Ecke aus A_G würde den Zusammenhang von G nicht zerstören.

□

Satz 5 (Gallai-Edmonds-Struktursatz).
Sei $G = (V, E)$ ein Graph. Dann gelten die folgenden Aussagen:

1. *Für alle $x \in A_G$ ist $D_{G-x} = D_G$ und $C_{G-x} = C_G$.*

2. *Jedes maximum matching in G enthält ein perfektes matching in $G[C_G]$ und maximum matchings jeder Komponenten von $G[D_G]$.*

3. *Jede Zusammenhangskomponente von $G[D_G]$ ist faktorkritisch.*

4. $\nu(G) = \frac{1}{2}(|V| + |A_G| - \kappa(G[D_G]))$.

Beweis:

1. „$\subseteq$“: Ein maximum matching M, welches $y \in D_G$ nicht überdeckt, enthält beim Entfernen von Kanten, die mit A_G inzidieren, keine neue Kante mehr, somit auch keine Kante mehr, die mit y inzidiert.

   „$\supseteq$“: Sei $x \in A_G$ und M ein maximum matching von $G-x$. Angenommen, $D_G \not\supseteq D_{G-x}$. Dann würde ein $y \in D_{G-x} \setminus D_G$ existieren. Sei M' das maximum matching in $G - x$, welches y vermeidet. Da $x \in A_G$, gibt es ein $z \in \Gamma(x) \cap D_G$ und somit auch ein maximum matching M in G, welches z vermeidet.

   Betrachte den Graphen $G' = (V, M \triangle M')$, wobei $M \triangle M' = (M \setminus M') \cup (M' \setminus M) = (M \cup M') \setminus (M \cap M')$. $E(G')$ besteht aus allen Kanten, die zu M und M' gehören ohne deren gemeinsamen Kanten. G' kann nur isolierte Punkte, alternierende Wege oder Kreise besitzen. y kann kein isolierter Punkt sein, da sonst beide inzidierenden Kanten zu beiden matchings oder zu keinem matching gehören. Da $y \notin D_G$, wird er von allen maximum matchings und insbesondere M überdeckt. Wäre die Komponente in G' von y ein alternierender Kreis, würde M' y nicht vermeiden. Sei also P die Komponente von y, das heißt, ein alternierender Weg.

   Dieser Weg kann aber ebenfalls nicht existieren. Der Nachweis erfolgt über folgende Teilaussagen, die jeweils zum Widerspruch geführt werden:

   (a) P hat gerade Länge.

   (b) P endet am Punkt x.

2. Wie in der intuitiven Erklärung.

3. Zeigen: H ein zusammenhängender Graph mit $\nu(H - x) = \nu(H)$ für alle $x \in V(H) \Longrightarrow$ H faktorkritisch.

   Dann ist auch jede Komponente H von $G[D_G]$ faktorkritisch, da es zu jedem $x \in V(H)$ ein maximum matching gibt, welches x vermeidet.

4.

□

Bedeutung der Aussagen auf intuitiver Ebene:

1. Die Punkte in D_G werden von einem maximum matching vermieden (aber nicht notwendig allen maximum matchings) und die Punkte in C_G werden von einem maximum matching überdeckt (aber nicht von allen maximum matchings). Diese Eigenschaften bleiben erhalten, wenn ein Punkt x entfernt wird, der zu jedem maximum matching gehört.

   $x \in D_G$ „Es gibt ein maximum matching, das x nicht überdeckt."
   $x \in A_G$ „Alle maximum matchings überdecken x und $x \in \Gamma(D_G)$."
   $x \in C_G$ „Alle maximum matchings überdecken x und $x \notin \Gamma(D_G)$."

   Zur Disjunktheit der Charakterisierungen: $x \notin D_G \iff X$ wird von jedem maximum matching überdeckt.

   A_G trennt D_G von C_G, das heißt, auf einem Weg zwischen D_G und C_G liegt notwendigerweise ein Punkt aus A_G.

2. Betrachte $G - A_G$ durch sukzessive entfernen der Punkte in A_G. Dann zerfällt G in $G[D_G]$ und $G[C_G]$.

   Die erste Aussage ist offensichtlich, da jeder Punkt in $G[C_G]$ von jedem maximum matching von G überdeckt wird. Dies ist in $G[D_G]$ nicht mehr der Fall, daher kann höchstens von maximum matchings in den Komponenten von $G[D_G]$ gesprochen werden. Diese sind aber bereits faktorkritisch, was in der nächsten Aussage bewiesen wird.

3. $x \in G[D_G]$

4.

Daraus können nun einige Folgerungen gezogen werden, die zum EDMONDS-Algorithmus führen, mit welchem ein maximum matching in allgemeinen Graphen in polinomieller Laufzeit gefunden werden kann.

Folgerung 2. *Sei $G = (V, E)$ ein Graph.*

1-Faktorsatz von TUTTE: *G besitzt genau dann ein perfektes matching, wenn $q(G - S) \leq |S|$ für alle $S \subseteq V$.*

BERGE-Formel: *Sei M ein maximales matching in G. Die Anzahl von M nicht überdeckter Punkte ist* $\max_{S \subseteq V}(q(G - S) - |S|)$.

Beweis:

1-Faktorsatz von TUTTE: „$\Rightarrow$": Betrachte $G - S$. Die ungeraden Komponenten müssen alle genau mit einer matching-Kante zu S verbunden gewesen sein, da sonst eine Ecke nicht vom matching überdeckt gewesen wäre und somit das matching nicht perfekt gewesen wäre. Das macht die Ungleichung $q(G - s) \leq |S|$ aus.

„$\Leftarrow$":

Berge-Formel: Für $S \subseteq V$ ist $q(G - S) - |S|$ genau die Anzahl der Punkte, die zu einem perfekten matching fehlen.

□

2.1.3 Netzwerke

Sei $N = (D, s, t, c)$ ein Netzwerk mit $D = (V, A)$.
Eine Folgerung aus dem Satz von Ford & Fulkerson ist das Min Cut Max Flow Theorem, also die Dualität zwischen maximalem Fluss und minimalem Schnitt.

Folgerung 3 (Min cut max flow). *In jedem Netzwerk ist der maximale Wert eines Flusses gleich der minimalen Kapazität eines Schnittes.*

Beweis: Wende Satz von Ford & Fulkerson an:
Wähle maximalen Fluss x aus dem Ford & Fulkerson-Algorithmus und minimalen Schnitt $C = \{v \in V \mid v = s \text{ oder ex. vergrößernder } (s - v)\text{-Weg}\}$ vermöge des Beweises zum Ford & Fulkerson-Algorithmus. Da $x(y, z) = c(y, z)$ für alle (y, z) mit $y \in C$ und $z \notin C$ gilt, ist auch $\text{cap}(C) = \text{val}(x)$. □

2.2 Matrizen

2.2.1 Doppelstochastische Matrizen

Lemma 3. *Seien $A, B \in \Omega_n$.*

- *$AB \in \Omega_n$.*
- *$A^T \in \Omega_n$ und somit $A^T A \in \Omega_n$.*

Beweis: Zum ersten Punkt:

$$\begin{aligned} (AB)_{ij} &= \sum_{k=1}^{n} a_{ik} b_{kj}, \\ \sum_{j=1}^{n} (AB)_j &= \sum_{j=1}^{n} \sum_{k=1}^{n} a_{ik} b_{kj} = \sum_{k=1}^{n} \sum_{j=1}^{n} a_{ik} b_{kj} = \underbrace{\sum_{k=1}^{n} a_{ik}}_{=1} \underbrace{\sum_{j=1}^{n} b_{kj}}_{=1} = 1, \end{aligned}$$

□

2.2.2 Permanente

Die Permanente kann in der Graphentheorie angewendet werden. Einige motivierende Beispiele liefert die folgende Bemerkung, die die im Folgenden aufgeführten Ergebnisse benutzt.

Bemerkung 8.
Sei $G = (V, E)$ ein Graph.

- $\mathrm{per}(A(G))$ *ist die Anzahl der perfekten matchings von* G.
- *Für* $A \in \mathbb{R}^{n\times n}$ *und* $k \in \mathbb{R}$ *ist* $\mathrm{per}(kA) = k^n \mathrm{per}(A)$.
- *Ist* G ***k*-regulär**, *das heißt,* $d(v) = k$ *für alle* $v \in V$, *so hat* G *mindestens* $k^n \frac{n!}{n^n}$ *perfekte matchings.*

Beweis:

- Ein Summand der Permanenten bezüglich einer Permutation $\pi \in S_n$ ist genau dann ungleich 0, wenn $\{a_{1,\pi(1)}, \ldots, a_{n,\pi(n)}\}$ nur positive Elemente enthält. Da $|\pi\{1, \ldots, n\}| = n$ ist und die Adjazenzmatrix symmetrisch ist, liegt genau dann auch ein perfektes matching vor.
- Es gilt nach Definition der Permanenten

$$\mathrm{per}(kA) = \sum_{\pi\in S_n} \prod_{i=1}^{n} k a_{i\pi(i)} = \sum_{\pi\in S_n} k^n \prod_{i=1}^{n} a_{i\pi(i)} = k^n \sum_{\pi\in S_n} \prod_{i=1}^{n} a_{i\pi(i)} = k^n \mathrm{per}(A).$$

- Nach der Van der Waeden-Vermutung ist $\mathrm{per}(A) \geq \frac{n!}{n^n}$, falls A doppelstochastisch ist. $\frac{1}{k}A(G)$ ist doppelstochastisch, somit ist

$$\mathrm{per}(A(G)) = k^n \mathrm{per}(\frac{1}{k}A(G)) \geq k^n \frac{n!}{n^n}.$$

□

Sei A eine $n \times n$-Matrix. Wie bei der Determinanten einer Matrix und dem Laplaceschen Entwicklungssatz kann auch die Permanente berechnet werden nur ohne die alternierenden Vorzeichen.

Satz 6.
(Entwicklungssatz für Permanenten)

$$\mathrm{per}(A) = \sum_{i=1}^{n} a_{ij} \mathrm{per}(A(i|j))$$

Beweis: Per Induktion über n wie bei der Determinanten. □

Permanenten sind zwar nicht unter Spalten- und Zeilentransformationen invariant wie etwa die Determinante, sondern nur unter Permutation von Zeilen und Spalten, was sich aus der Definition der Permanenten sofort ergibt.

Bemerkung 9.
Sei $(a_{ij}) = A \in \mathbb{R}^{n\times n}$.

- *Sein* $a_{ij} \geq 0$. *Es gilt:*

$$\operatorname{per}(A) = 0 \quad \Longleftrightarrow \quad A[I|J] = 0 \in \mathbb{R}^{k\times l} \text{ mit } |I| = k, |J| = l.$$

- *Ist* $A \in \Omega_n$ *zerlegbar, so ist* A *durch Zeilen- und Spaltenpermutationen auf die Form* $A = \begin{pmatrix} B & 0 \\ C & 0 \end{pmatrix}$ *bringbar, wobei* B, C *quadratische Matrizen sind.*
- *Ist* $A \in \Omega_n$ *unzerlegbar, so ist* $\operatorname{per}(A(i|j)) > 0$ *für alle* $1 \leq i, j \leq n$ *also auch* $\operatorname{per}(A) > 0$.
- *Ist* A *minimierend, so ist* A *unzerlegbar.*

Beweis:

-
- Klar: Die Zeilensummen von B und die Spaltensummen von C ergeben jeweils stets 1. Also können die Zeilenelemente in B respektive die Spaltenelemente in C jeweils so permutiert werden, dass auch die Spaltensummen von B respektive die Zeilensummen von C jeweils stets 1 ergeben. Damit sind B und C bereits doppelstochastisch und D ergibt sich zwingend zu 0.
-
- Angenommen A wäre zerlegbar. Wird in der mittleren 2×2-Untermatrix von A ein ϵ auf jeden Eintrag addiert, so ergibt sich die Permanente dieser Matrix zu einem kleineren Wert als dem der Permanente von A im Widerspruch zur Minimalität von A.

□

Van der Waeden Vermutung

Die Verwendung der Permanenten in der Optimierung geht hauptsächlich auf Satz 8 zurück. Zum Beweis dieser Vermutung werden einige Begriffe und Ergebnisse benötigt, die hier nun entwickelt werden. Ziel ist es, zu zeigen, dass das kleinste Element bezüglich der Permanenten, eine minimierende Matrix, die Van der Waeden Vermutung erfüllt.

Lemma 4. *Sei $A \in \Omega_n$ minimierend.*

- *Ist $a_{ij} > 0$, so ist* $\mathrm{per}(A) = \mathrm{per}(A(i|j))$.
- *Ist $a_{ij} = 0$, so ist* $\mathrm{per}(A(i|j)) \geq \mathrm{per}(A)$.

Beweis: *Minimiere Matrix bezüglich der Permanenten unter Verwendung von* Lagrange-*Multiplikatoren.* □

Satz 7 (Alexandrov-Fenchel Ungleichung).
Seien $(a_1, \ldots, a_{n-2}) = A \in \mathbb{R}^{n\times(n-2)}$ mit $a_{ij} \geq 0$ und $x \in \mathbb{R}^n$ mit $x_i \geq 0$.
Dann gilt für $y \in \mathbb{R}^n$:

$$\mathrm{per}((A, x, y))^2 \geq \mathrm{per}((A, x, x))\mathrm{per}((A, y, y))$$

Beweis: □

Satz 8 (Van der Waeden Vermutung).
Ist $A \in \Omega_n$, so ist $\mathrm{per}(A) \geq \frac{n!}{n^n}$ *mit Gleichheit genau dann, wenn $A = (\frac{1}{n})_{1\leq i,j\leq n}$.*

Beweis:[Egorychev, Falikman, 1980] Benutzt Lemma 4 und die Alexandrov Fenchel Ungleichung. □

Eine wichtige Tatsache ist die genaue Charakterisierung der minimierenden Matrizen.

Folgerung 4. *Sei $A \in \Omega_n$.*

$$\mathrm{per}(A) = \tfrac{n!}{n^n} \iff A = \begin{pmatrix} \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \ddots & \vdots \\ \frac{1}{n} & \cdots & \frac{1}{n} \end{pmatrix} \iff A \text{ ist minimierend.}$$

2.3 Ganzzahlige lineare Optimierung

Das lineare Optimierungsproblem aus Abschnitt 1.3 wird nun in ein ganzzahliges lineares Optimierungsproblem (Integer Linear Programme) umgeformt. Die Abweichung ist die Ganzzahligkeit der Lösung.
Seien $A \in \mathbb{R}^{m\times n}$, $b, c \in \mathbb{R}^m$.

Problem 3 (ILP).

$$\begin{aligned} \max \quad & c^T x \\ Ax \quad &\leq \quad b \\ x \quad &\geq \quad 0, \qquad \mathbf{x} \in \mathbb{Z} \end{aligned}$$

Im Allgemeinen ist ILP $\mathcal{NP}$-vollständig, wie etwa durch Reduktion von 3-SAT gezeigt werden kann. Im Folgenden wird in erster Linie die Effektivität in den Vordergrund gestellt.
Die Existenz einer ganzzahligen Lösung kann durch verschiedene Aspekte charakterisiert werden, wozu die folgenden Begriffe benötigt werden. Die Vorgehensweise dabei ist zunächst

- die Charakterisierung der Eckpunkte des Lösungspolyeders als minimale Strata,
- die Herleitung von Bedingungen für die Ganzzahligkeit der minimalen Strata.

Definition und Bemerkung 15.

- $A \in \mathbb{Z}^{n\times n}$ *heißt* ***unimodular****, falls* $\det(A) \in \{\pm 1\}$.
- $A \in \mathbb{Z}^{m\times n}$ *heißt* ***vollständig unimodular****, falls* $\det(A') \in \{0, \pm 1\}$ *für jede quadratische Untermatrix* A' *von* A.

1. *Ist* $A \in \mathbb{Z}^{n\times n}$ *vollständig unimodular und* $\det(A) \neq 0$*, so ist* A *auch unimodular.*
2. $A \in \{0, \pm 1\}^{m\times n}$*, falls* A *vollständig unimodular ist.*
3. *Sei* L *ein volles Gitter in* $\mathbb{R}^n$*, also* $\dim L = n$*. Ist* L *unimodular, so auch* $\mathcal{G}(L)$.
4. *Ist* A *vollständig unimodular, so auch* A^T, $-A$, $[A|-A]$, $\begin{pmatrix} A \\ I \end{pmatrix}$.

Beweis:

1. $\det(A')\{\pm 1\}$ für jede quadratische Untermatrix. Mit der La Placeschen Entwicklung ergibt sich somit ein Produkt von ± 1en.
2. $A \in \mathbb{Z}^{m\times n}$ vollständig unimodular und $A' \in \mathbb{Z}^{r\times r}$ eine Untermatrix von A mit $r \leq \min\{m, n\}$. Nach dem La Placeschen Entwicklungssatz für Determinanten ist $A' \in \{0, \pm 1\}^{r\times r}$, also auch A.

   Die Umkehrung gilt nicht, da beispielsweise $A = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$ Determinante 2 hat.
3. (Siehe auch [LoupGC].) $L = L^{\#} \Longrightarrow \det(L) = \det(L^{\#})$. Wegen $\det(L) = \det(\mathcal{G}(L))$ und $\det(\mathcal{G}(L))\det(\mathcal{G}(L^{\#})) = 1$ folgt $\det(L) \in \{\pm 1\}$ also $\mathcal{G}(L)$ unimodular.
4. Klar, da sich die Einträge nicht wesentlich ändern.

□

Um den **Lösungsraum** des ILP (Problem 3)

$$M_b := \{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0\},$$

der geometrisch ein Polyeder ist, zu charakterisieren, sind folgende Begriffe nötig.

Definition und Bemerkung 16.

- *Für $\overline{x} \in M_b$ definiere $J_0(\overline{x}) := \{j \in \{1, \ldots, m\} \mid a_j\overline{x} = b_j\}$ als die Menge der Zeilennummern von A, so dass deren Zeile der Ungleichung $A\overline{x} \leq b$ aktiv ist: $j \in J_0(\overline{x}) \Longrightarrow a_j\overline{x} = b_j$.*
- $\Sigma(\overline{x}) := \{x \in M_b \mid J_0(x) = J_0(\overline{x})\} \subseteq M_b$ *heißt* ***Stratum von*** $\overline{x}$.
- $\text{aff}(a_1, \ldots, a_k) := \{x \in \mathbb{R}^n \mid$ *ex.* $\lambda_1, \ldots, \lambda_k : \sum_i \lambda_i = 1, \sum_i \lambda_i a_i = x\}$ *heißt der* ***affine Unterraum von*** $a_1, \ldots, a_k \in \mathbb{R}^n$
- $\text{aff}(\Sigma(x))$ *ist der kleineste affine Unterraum, in dem* $\Sigma(x)$ *liegt und heißt die* ***affine Hülle von*** $\Sigma(x)$.
- $\dim \Sigma\ (x) := \dim \text{aff}\ (\Sigma(x))$
- *Ein Stratum* $\Sigma(x)$ *heißt* ***minimal****, falls* $\dim \Sigma\ (x)$ *minimal ist.*

1. $\text{aff}(a_1, \ldots, a_k)$ *ist die Menge aller Konvexkombinationen der* $a_1, \ldots, a_k \in \mathbb{R}^n$.
2. $\text{aff}(a_1, \ldots, a_k) \leq \mathbb{R}^n$
3. $\dim \Sigma\ (x) = n - |J_0(x)|$.
4. $\Sigma(x)$ *ist genau dann minimal, wenn* $\{a_j \mid j \in J_0\} = \text{Zr}(A)$.
5. *Minimale Strata sind affine Unterräume.*

Beweis:

1. Klar.
2. Seien $x, y \in \text{aff}(a_1, \ldots, a_k)$, $x = \sum_i \lambda_i a_i$ und $y = \sum_i \lambda'_i a_i$. $\Longleftrightarrow$

$$x + y = \sum_i \lambda_i a_i + \sum_i \lambda'_i a_i = \sum_i (\lambda_i + \lambda'_i) a_i \in \text{aff}(a_1, \ldots, a_k).$$

3.
4.

□

Falls a_j die j-te Spalte von A ist, definieren die Ungleichungen $a_j x \leq b_j$ jeweils einen Halbraum von $\mathbb{R}^n$, der durch die Hyperebene definiert durch $a_j x = b_j$ begrenzt ist. Der Schnitt dieser Halbräume beschreibt das Lösungspolyeder M_b, dessen Grenzflächen die Hyperebenen vermöge $a_j x = b_j$ sind. Die Zielfunktion $c^T x$ verläuft durch dieses Polyeder.
Sind die Eckpunkte ganzzahlig, so heißt M_b ganzzahliges Polyeder (siehe Definition 17).

Sei $x \in M_b$. $J_0(x)$ die Menge der Zeilennummern, deren entsprechenden Nebenbedingungen aktiv sind, das bedeutet, alle y, die ebenfalls zu $J_0(x)$ gehören, erfüllen genau die gleichen Nebenbedingungen, liegen also in dem offenen Inneren der jeweiligen Hyperebene.
Diese Punkte mit gleicher Nebenbedingung-Konfiguration bilden das Stratum $\Sigma(x)$ zu einem ausgezeichneten Punkt x des Lösungsraums.
Minimale Strata sind die Strata $\Sigma(\overline{x})$ mit

$$\dim \Sigma\ (\overline{x}) = \min_{x \in M_b} \dim \Sigma\ (x).$$

Das bedeutet, minimale Strata sind die affinen Unterräume von M_b mit der kleinsten Dimension. Im Falle eines Polyeders sind das also die Eckpunkte. Somit ist eine Charakterisierung für die möglichen optimalen Lösungen vermöge der Zielfunktion gefunden, da, falls ein Optimum existiert, dies auch auf einem Eckpunkt des Lösungspolyeders angenommen werden kann.

Beispiel 2. *Sei M ein Würfel im $\mathbb{R}^3$, somit $n = 3$.*

- *Betrachte die Punkte auf einer Seite eines Würfels. Sobald die Seite mit einer anderen zusammenläuft, entsteht ein Eckpunkt, auf dem andere Nebenbedingungen aktiv sind und J_0 ist eine andere Menge als im Inneren einer der Seiten. $|J_0(x)|$ ist die Anzahl der gerade aktiven Nebenbedingungen.*

- *Strata von M und deren Eigenschaften*

Strata $\Sigma(x)$:	*Eckpunkte*	*offene Seiten*	*offene Seitenflächen*	$\overset{\circ}{M}$
$\mathrm{aff}(\Sigma(x))$*:*	*Punkte*	*Geraden*	*Ebenen*	$\mathbb{R}^3$
$\dim \Sigma\ (x)$*:*	0	1	2	3
$\lvert J_0(x)\rvert$*:*	3	2	1	0

Die minimalen Strata von M sind demnach die Eckpunkte von M.

Die Ganzzahligkeit der minimalen Strata bedeutet damit die Ganzzahligkeit der optimalen Lösungen, von denen eine ja ein Eckpunkt beziehungsweise ein Stratum des Lösungspolyeders ist.

Satz 9.

1. *Ist $A \in \mathbb{R}^{m \times n}$ vollständig unimodular, so enthält jedes minimale Stratum von M_b für jedes $b \in \mathbb{Z}^m$ einen ganzzahligen Punkt.*

2. *Genau dann ist $A \in \mathbb{Z}^{m \times n}$ vollständig unimodular, wenn die Ecken von M_b für jedes $b \in \mathbb{Z}^m$ ganzzahlig sind.*

Beweis:

□

Unimodulare Matrizen sind also für die Ganzzahligkeit der Lösungen verantwortlich. Um nun eine weitere Bedingung herzuleiten, wird das duale Problem herangezogen.

Erinnerung 2.

$$\begin{array}{llll} (P): & \max & c^T x & \\ & Ax & \leq & b \\ & x & \geq & 0 \end{array} \qquad \begin{array}{llll} (D): & \min & y^T b & \\ & y^T A & \geq & c^T \\ & y & \geq & 0 \end{array}$$

Nach dem starken Dualitätsprinzip ist $\max\{c^T x \mid Ax \leq b, x \geq 0\} = \min\{y^T b \mid y^T A = c^T, y \geq 0\}$.

Satz 10.

1. *Ist $A \in \mathbb{Z}m \times n$, so existieren unimodulare Matrizen U, V, so dass*

$$UAV = \operatorname{diag}(d_1, \ldots, d_s, 0, \ldots, 0)$$

   mit $d_i > 0$ und $d_i \mid d_{i+1}$ für alle $0 \leq i \leq s - q$.

2. *$A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Q}^m$. Das Gleichungssystem $Ax = b$ ist genau dann lösbar mit $x \in \mathbb{Z}^n$, wenn für jedes $y \in \mathbb{Q}^m$ mit $x^T A \in \mathbb{Z}^n$ auch $y^T b \in \mathbb{Z}$.*

Beweis:

1. Elementarteileralgorithmus.
2. „⇒“: Sei $x \in \mathbb{Z}^n$. Es gilt $y^T(\underbrace{Ax}_{\in \mathbb{Z}^m}) = y^T b$ und falls $y^T A \in \mathbb{Z}^n$, so auch $y^T b$.

   „⇐“: Verwende die Zerlegung $\operatorname{diag}(d_1, \ldots, d_s, 0, \ldots, 0) = D = UAV$.

□

Um nun auch eine praktische Lösbarkeit zu erreichen, wird noch ein Ansatz für die Bestimmung der Unimodularität von Matrizen benötigt.

Satz 11 (Ghonila-Houri).
$A \in \mathbb{Z}^{m \times n}$ ist genau dann vollständig unimodular, wenn für alle $J \in \{1, \ldots, n\}$ eine Zerlegung $J = J_1 \uplus J_2$ existiert, so dass

$$\sum_{j \in J_1} a^j - \sum_{j \in J_2} a^j \in \{0, \pm 1\}$$

oder wegen Invarianz unter Transponation

$$\sum_{i \in J_1} a_i - \sum_{i \in J_2} a_i \in \{0, \pm 1\}.$$

Ein weiterer Ansatz zur Formulierung der Ganzzahligkeit der Lösungen des ILPs sind die **ganzzahligen Polyeder**.

Definition 17. *Sei M ein Polyeder.*

- *M heißt **ganzzahlig**, falls $M = \mathcal{C}(M \cap \mathbb{Z}^n) =: M_{int}$.*[19]
- *M heißt **rational**, falls es $A \in \mathbb{Q}^{m \times n}$ und $b \in \mathbb{Q}^m$ gibt mit $M = \{x \in \mathbb{R}^n \mid Ax \leq b\}$.*

Bemerkung 10.

- *Ist M ein rationales Polyeder, so ist M_{int} auch rational.*
- *M ist genau dann ein ganzzahliges Polyeder, wenn $M = \mathcal{C}(a_1, \ldots, a_k) + \mathcal{K}(b_1, \ldots, b_r)$ für $a_i, b_i \in \mathbb{Z}$, wobei $\mathcal{K}(b_1, \ldots, b_r) := \{x \in \mathbb{R}^n \mid x = \sum_{i=1}^{k} a_i b_i, a_i \geq 0\}$ der von $b_1, \ldots, b_r$ **aufgespannte Kegel** ist.*

Beweis:

□

Satz 12.

1. *M ist ganzzahlig.*
2. *Jedes minimale Stratum enthält einen ganzzahligen Punkt.*
3. *Für $c \in \mathbb{Z}^n$ oder sogar $c \in \mathbb{R}^n$ ist $\max\{c^T x \mid x \in M\} \in \mathbb{Z}$, sofern das Maximum existiert.*

Beweis:

1. „⇐“: Die Eckpunkte von M sind ganzzahlig, die optimale Lösung kann also ganzzahlig gewählt werden.

   „⇒“: Wird das Maximum auf einem ganzzahligen Punkt angenommen, so befindet sich dieser am Rand von M.

□

Ein weiterer Ansatz zur Charakterisierung der Ganzzahligkeit von Lösungen des ILPs ist die **totale duale Ganzzahligkeit** eingeführt von Edmonds und Giles 1977.

Definition 18. *Ein Ungleichungssystem $Ax \leq b$ heißt **total dual ganzzahlig** (TDI)*[20]*, falls*

$$\min\{y^T b \mid y^T A = c^T, y \geq 0\} \in \mathbb{Z}$$

für jedes $c \in \mathbb{Z}^n$, sofern das Minimum existiert.

Ist $Ax \leq b$ TDI, so kann daraus nicht direkt die Ganzzahligkeit der Lösung des zugehörigen ILPs gefolgert werden. Dazu sind noch Zusatzannahmen nötig.

Bemerkung 11.

Sei (P) lösbar. Ist $Ax \leq b$ TDI, so existiert eine ganzzahlige Lösung von (P).

[19] Das bedeutet, alle Eckpunkte sind ganzzahlig.

[20] Englisch: totally dual integre

Beweis: Es sei $\overline{y}^T b := \min\{y^T b \mid y^T A = c^T, y \geq 0\} = \max\{c^T x \mid Ax \leq b, x \geq 0\} =: c^T \overline{x}$. Es ist $\overline{y}^T b = c^T \overline{x} \in \mathbb{Z}$ und wegen $c \in \mathbb{Z}^n$ auch $\overline{x} \in \mathbb{Z}^n$. □

Satz 13 (Giles, Pulleyblank).
Ist M ein rationales Polyeder, dann existiert ein TDI System $Ax \leq b$, so dass $M = \{x \mid Ax \leq b\}$.

Beweis: Hilbert-*Basis bestimmen.* □

Zusammenfassung:

Folgerung 5. *Sei das primale Problem (P) für alle $c \in \mathbb{Z}^n$ lösbar, das bedeutet, $\max\{c^T x \mid x \in M_b\} < \infty$ und sei $M = \{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0\}$, sowie das duale Problem (D) für alle $b \in \mathbb{Z}^m$ lösbar.*
Genau dann gibt es eine ganzzahlige Lösung $\overline{x}$ von (P), wenn eine der folgenden Bedingungen gilt.

1. *M ist ganzzahlig.*
2. *Jedes minmale Stratum enthält einen ganzzahligen Punkt.*
3. *$A \in \mathbb{Z}^{m \times n}$ ist vollständig unimodular.*
4. *$Ax \leq b$ ist TDI.*

Beweis:

1. Nach Satz 12.
2. Nach Satz 12.
3. Nach Satz 9 ist M genau dann ganzzahlig, wenn $A \in \mathbb{Z}^{m \times n}$ vollständig unimodular und ganzzahlig ist.
4. Nach Bemerkung 11.

□

2.4 Komplexitätstheorie

Um die Komplexitätstheorie für die Klasse $\mathcal{NP}$ mittels des Reduktionsbegriffes für Probleme aufbauen zu können, bedarf es eines ursprünglichen Problems, welches $\mathcal{NP}$-vollständig ist und auf das weitere Probleme reduziert werden können. Dieses Problem muss auf alle Probleme in $\mathcal{NP}$ reduziert werden, was im Beweis des folgenden Satzes getan wird.

Satz 14 (Satz von Cook).
SAT *ist $\mathcal{NP}$-vollständig.*

Beweis:

1. SAT $\in \mathcal{NP}$: Prüfe die Eingabe auf Zulässigkeit, *rate* eine erfüllende Belegung für die Formel und verifiziere diese.

2. Sei $L \in \mathcal{NP}$ bliebig. M_L sei der nichtdeterministische Akzeptor von L.

   Zeige: M_L kann durch in polynomiellem Aufwand durch eine Formel dargestellt werden, die genau dann erfüllbar ist, wenn M_L die Eingabe akzptiert.

   Idee: Beschreibe zunächst die Konfigurationen durch Formeln und stelle dann den Übergang zu einer Folgekonfiguration dar.

□

Die Hauptergebnisse werden im folgenden Satz zusammengefasst.

Satz 15.

1. KNF-SAT *ist $\mathcal{NP}$-vollständig.*

2. 3-SAT *ist $\mathcal{NP}$-vollständig.*

3. Folgende Polynomzeit-Reduktionen gelten:

- 3-SAT $\overset{3.}{\propto}$ XC $\overset{4.}{\propto}$ 3-DM $\overset{5.}{\propto}$ HC
- 3-DM $\overset{6.}{\propto}$ CLIQUE $\overset{7.}{\propto}$ INDEPENDENT SET $\overset{8.}{\propto}$ VC

Die Beweise und die entsprechenden Probleme werden im Folgenden stückweise aufgeführt.

1.

Problem 4 ((KNF-)SAT).

Gegeben:	*Aussagenlogische Formel ϕ (in konjungtiver Normalform).*
Frage:	*Ist ϕ erfüllbar?*

Bemerkung 12.
Jede aussagenlogische Formel ϕ ist in Polynomzeit auf eine Formel ϕ' in KNF überführbar.

Die Umwandlung der SAT-Eingabeformel in KNF ergibt eine Reduktion von SAT auf KNF-SAT.

Folgerung 6. KNF-SAT *ist $\mathcal{NP}$-vollständig.*

2.

Problem 5 (3-SAT).

Gegeben:	*Aussagenlogische Formel ϕ in KNF mit drei Literalen pro Klausel.*
Frage:	*Ist ϕ erfüllbar?*

Satz 16.
3-SAT *ist $\mathcal{NP}$-vollständig.*

Beweis: Reduktion von KNF-SAT.

Idee: Konstuiere erfüllbarkeitsäquivalente Klauselmenge, deren Klauseln immer nur drei Literale enthalten. Starte mit zwei Literalen einer Klausel und nehme ein Literal aus einer anderen Klausel hinzu. Die nächste Klausel wird aus zwei alten und wieder einem neuen Literal gebildet und so weiter.

□

3.

Problem 6 (XC (EXACT COVER)).

Gegeben:	*Endliche Menge S und ein Mengensystem $\mathcal{C} \subseteq \mathcal{P}\mathrm{ot}(S)$.*
Frage:	*Existiert eine Teilmenge $C \subset \mathcal{C}$, so dass*
	1. $\bigcup C = S$,
	2. $E, F \in C \Rightarrow E \cap F = \emptyset$?

Satz 17.
XC *ist $\mathcal{NP}$-vollständig.*

Beweis: Reduktion von 3-SAT.

Idee: Wähle $\mathcal{C} = \{C_1, C_2, C_3\}$ so, dass die C_i Mengen von Literalen enthalten, die genau so gewählt wurden, dass die Mengen bei einer erfüllenden Belegung der Variablen eine exakte Überdeckung besitzen. □

4.

Problem 7 (3-DM (3-DIMENSIONAL MATCHING)).

Gegeben:	*Drei disjunkte Mengen X, Y, Z alle der Kardinalität n*
	Mengensystem $\mathcal{E} \subseteq \mathcal{P}\mathrm{ot}(X \uplus Y \uplus Z)$, so dass
	$\lvert X \cap E\rvert = \lvert Y \cap E\rvert = \lvert Z \cap E\rvert = 1$ für alle $E \in \mathcal{E}$.
Frage:	*Enthält $\mathcal{E}$ eine exakte Überdeckung wie in XC?*

Satz 18.
XC *ist $\mathcal{NP}$-vollständig.*

Beweis: Reduktion von XC.

Idee: Wähle aus je einer Menge S der Überdeckung je drei Elemente so, dass eines nur in S und die beiden anderen im Schnitt mit je einer anderen Menge der Überdeckung sind. Diese drei Elemente bilden dann zusammen eine Menge mit der gewollten Eigenschaft.
□

5.

Problem 8 (HC (HAMILTONIAN CIRCUIT)).

Gegeben:	*Graph $G = (V, E)$.*
Frage:	*Besitzt G einen* HAMILTON*-Kreis?*

Satz 19.
HC *ist $\mathcal{NP}$-vollständig.*

Beweis: Reduktion von 3-DM.

Idee: Die Elemente der Dreiermengen können auf eindeutige Weise durchlaufen werden.
□

6.

7.

8.

3 Probleme und deren Lösung

In diesem Abschnitt werden konkrete Problemstellungen vorgestellt und Lösungsansätze mittels der in Abschnitt 2 hergeleiteten Ergebnisse entwickelt.

3.1 Minimaler Spannbaum

Problem 9 (MINIMUM SPANNING TREE).

Gegeben:	*Graph $G = (V, E)$, Kostenfunktion $c : E \to \mathbb{R}$.*
Gesucht:	*MST von G.*

Ein MST kann mittels folgendem Greedy-Algorithmus[21] berechnet werden, der auf zusammenhängenden Graphen arbeitet. Dies ist keine Einschränkung für obiges Problem, da bei einem nicht-zusammenhängenden Graphen kein MST existiert.

Algorithmus 1 (KRUSKAL-Algorithmus).

Eingabe:	*$G = (V, E)$ zusammenhängend, $c : E \to \mathbb{R}$.*
(1)	*Sortiere E nach Bewertung $c(e)$ für alle e,*
	so dass $E = \{e_1, \ldots, e_m \mid c(e_i) \leq c(e_j)$ für $i < j\}$.
	Setze $E_0 := \emptyset$, $i := 1$.
(2)	*Falls $E_i \cup \{e_i\}$ einen Kreis enthält, setze $E_{i+1} := E_i$ und $i := i + 1$.*
	Sonst setze $E_{i+1} := E_i \cup \{e_i\}$.
	Setze $i := i + 1$ und mache weiter bei (2), falls $i < m$, sonst stoppe.
Ausgabe:	*MST (V, E_m).*

Satz 20 (Korrektheit des KRUSKAL-Algorithmus).
Die Ausgabe des KRUSKAL*-Algorithmus ist ein MST.*

Beweis: Seien $G = (V, E)$ der zu Grunde liegende Graph und O.B.d.A. $E = \{e_1, \ldots, e_m \mid c(e_i) \leq c(e_j)$ für $i < j\}$ sowie T die Ausgabe des KRUSKAL-Algorithmus.

1. T ist ein Graph.

   Nach Konstruktion ist $T \subseteq G$ ein Untergraph von G.

2. T ist ein Baum.

   (a) Da nie Kreise geschlossen werden, ist T ein Wald.

   (b) Angenommen, T wäre nicht zusammenhängend. Sei $V_1 \uplus V_2$ die Partition von V, so dass $E(T) \cap V_1 \times V_2 = \emptyset$. In G gibt es Kanten zwischen V_1 und V_2, sei e_i die Kante mit minimalem $c(e_i)$. Da e_i keinen Kreis schließt, wählt der Algorithmus diese Kante. Widerspruch.

3. T ist ein MST.

   Angenommen T wäre nicht minimal. Wähle einen MST T' von G mit $E(T) \cap E(T')$ maximal. Sei e_k die erste[22] Kante aus $E(T) \setminus E(T')$, das heißt, die erste Kante, die der Algorithmus abweichend von T' wählt. Dann enthält $T' + e_k$ einen Kreis C, da T' sonst nicht alle Punkte von G enthielte. Nun gibt es ein $e_l \in E(C) \setminus E(T)$ in $T' + e_k$, da sonst C bereits vollständig in T liegen würde. Daher ist $l > k$. Tausche nun e_k mit e_l und

[21] Das bedeutet, in jedem Schritt wird immer nur der beste Wert bezüglich der gewählten Heuristik verwendet (hier der niedrigste c-Wert).

[22] bezüglich der Sortierung der Kanten

erhalte den Spannbaum $T'' := (T' + e_k) - e_l$. Also gilt $c(e_l) \geq c(e_k)$. Die Kosten von T'' ergeben sich nun zu

$$c(T'') = c((T' + e_k) - e_l) = c(T') + \underbrace{c(e_k) - c(e_l)}_{\leq 0} \leq c(T').$$

Da T' aber minimal ist, muss $c(T'') = c(T')$ sein und T'' ein MST mit einer mit T gemeinsamen Kante mehr nämlich e_k. Dies ist ein Widerspruch zur Wahl von T', der bereits maximal mit dieser Eigenschaft war, weswegen T MST ist.

□

3.2 Matching Probleme

Der erste Schritt im algorithmischen Umgang mit bipartiten Graphen ist das Feststellen, ob der Graph tatsächlich bipartit ist. Dies kann jedoch effizient bewerkstelligt werden.

Problem 10 (BIPARTITE GRAPH).

Gegeben:	*Graph* $G = (V, E)$.
Gesucht:	*Bipartition* $V = U \uplus W$ *mit*
	$x \in U$ *und* $y \in W$ *oder* $y \in U$ *und* $x \in W$ *für alle* $xy \in E$.

Das Werkzeug zur Bestimmung der Bipartition ist der Begriff der Färbbarkeit des Graphen.

Definition und Bemerkung 19. *Sei* $G = (V, E)$ *ein Graph.*

- *Eine Abbildung* $c : V \to \{1, \ldots, k\}$ *mit* $1 \leq k \in \mathbb{N}$ *heißt* ***k-Färbung****.* c *heißt* ***zulässig****, falls* $c(x) \neq c(y)$ *für alle* $xy \in E$. G *heißt* ***k-färbbar****, falls eine zulässige* k*-Färbung für* G *existiert.*

  $\chi(G) := \min\{k \in \mathbb{N} \mid G \text{ besitzt eine zulässige } k\text{-Färbung}\}$

- *Genau dann ist* $\chi(G) \leq 2$*, wenn* G *bipartit ist.*

Beweis: „⇒ "“: Sei $f := \chi(G) \leq 2$ vermöge der $\chi(G)$-Färbung c. Ist $f = 1$, so ist $E = \emptyset$ und somit $V = V \uplus \emptyset$ Bipartition. Sei nun $f = 2$. Wähle Bipartition $V = U \uplus W$ mit $U := c^{-1}(1)$ und $W := c^{-1}(2)$. Sei $xy \in E$. Ist $x \in U$, so ist $y \in W$, da $1 = c(x) \neq c(y) = 2$. Umgekehrt gilt $x \in W \Longrightarrow y \in U$.
„⇐“: Sei nun G bipartit mit Bipartition $V = U \uplus W$. Setze $c(U) := 1$ und $c(W) := 2$. Da keine Kanten innerhalb von U beziehungsweise W verlaufen, ist G auch 2-färbbar.

□

Der Algorithmus zum Testen, ob ein Graph bipartit ist, weist also die $\leq$ 2-Färbbarkeit des Graphen nach wie folgt. Der Einfachheit halber wird ein zusammenhängender Graph betrachtet, der Algorithmus kann aber leicht durch Anwendung auf die verschiedenen Komponenten auf allgemeine Graphen angewandt werden.

Algorithmus 2 (Bipartitionstest).

Eingabe:	*$G = (V, E)$ zusammenhängend.*
(0)	*Wähle $v_0 \in V$.*
(1)	*Für jedes i:*
	Sei $c(v_i) = f \in \{1, 2\}$ und $\{\bar{f}\} = \{1, 2\} \setminus \{f\}$.
	Falls $f \in c(N(c_i))$, stoppe mit Ausgabe „Nein“.
	Sonst setze $c(N(c_i)) := \bar{f}$.
Ausgabe:	*Bipartition $V = c^{-1}(1) \uplus c^{-1}(2)$.*

Diesen speziellen matching Problemen liegt der Satz von König zu Grunde (Satz 3), der die Dualität zwischen maximum matching und minimum vertex cover beschreibt. Im Folgenden wird deutlich, dass bipartite matchings algorithmisch noch in polynomiellem Aufwand bestimmbar sind, also eine bewältigbare Klasse von Problemen darstellen.

Problem 11 (BIPARTITE MATCHING).

Gegeben:	*Bipartiter Graph $G = (U \uplus W, E)$.*
Gesucht:	*maximum matching oder minimum vertex cover in G.*

Die Konstruktion eines solchen matchings beziehungsweise vertex covers leistet der folgende Algorithmus (vergleiche Satz 21). Dazu werden einige Vorbemerkungen getroffen.

Definition und Bemerkung 20. *Sei $G = (U \uplus W, E)$ ein bipartiter Graph und M ein matching in G.*

- *Ein M-**alternierender Weg** ist ein Weg $x_0 e_1 x_1 \dots e_r x_r$, so dass $e_i \in M$ und $e_{i-1}, e_{i+1} \notin M$ für alle i.*
- *M kann mittels eines M-alternierenden Weges $P = x_0 e_1 x_1 \dots e_r x_r$ vergrößert werden, sofern $e_1, e_r \notin M$. In diesem Fall heißt P auch M-**erweiternder** Weg.*

Beweis: Aus dem M-alternierenden Weg $P = x_0 e_1 x_1 \dots e_r x_r$ mit $e_1, e_r \notin M$ konstruiere erweitertes matching M' durch Entfernen der $e_i \in M$ und Hinzufügen der $e_i \notin M$. Dies kommt der symmetrischen Differenz von M und $E(P)$ gleich: $M' = M \triangle E(P) = \underbrace{(M \setminus E(P))}_{\text{Entfernen}} \cup \underbrace{(E(P) \setminus M)}_{\text{Hinzufügen}}$. □

Algorithmus 3 (Ungarische Methode).

Eingabe:	$G = (U \uplus W, E)$ *bipartit, initiales matching* $M \subseteq E$,
	$X_1 = \{x \in X \mid x \notin e \text{ für alle } e \in M\}, \quad X \in \{U, W\}$
(1)	Markierung
(1.0)	*Jeder Punkt aus* U_1 *wird mit „0“ markiert.*
(1.1)	*Wähle den nächsten, nicht durchgesehenen, markierten Punkt* v.
	Falls $v \in U$*:*
(1.2)	*Für jede Kante* $vw \in E \setminus M$ *markiere alle nicht markierten* w *mit „*v*“.*
(1.3)	*Falls* $v \in W$*:*
	Falls $v \in W_1$, $\rightarrow$ *(2).*
	Sonst wähle Kante $vu \in M$ *und markiere* u *mit „*v*“.*
	Definiere v *als „durchgesehen“.*
(2)	Erweiterung von M
	Konstruiere M*-erweiternden Weg* $P = v_0 e_1 v_1 \dots v_{r-1} e_r e_r$*, wobei*
	$v_0 = v \in W_1$ *und* $v_r \in U_1$ *(Marke „0“) sowie*
	$v_i \in U$ *genau dann, wenn* $v_{i-1} \in W$ *für* $1 \leq i \leq r$*, indem der jeweils*
	nächste Punkt durch die Marke des aktuellen Punktes bestimmt ist.
	Setze $M := M \triangle E(P)$.
	Lösche alle Knoten und „durchgesehen“-Marken. $\rightarrow$ *(1).*
Ausgabe:	
MM:	M
MVC:	*Markierte Punkte in* $W \cup$ *nicht markierte Punkte in* U

Bemerkung 13.

- *Ist ein* M*-erweiternder Weg gefunden, der mit einem Knoten* u *mit Marke „0“ endet, so werden nach Bestimmen der symmetrischen Differnez auch die Marken „0“ gelöscht, so dass nach Schritt (1.0)* u *nicht mehr mit „0“ markiert ist.*
- *Wird mit* $M = \emptyset$ *gestartet oder einem großen Graphen, der nur wenig von* M *überdeckt ist, so bestehen die ersten Schritte stets aus Hinzufügen einzelner Kanten zu* M*, da noch keine Wege konstruiert werden können.*
- *Das merken des Zustands „durchgesehen“ kann durch eine weitere Marke oder eine Liste implementiert werden.*

In dem Algorithmus werden zu jedem Punkt $v \in U$ die Nachbarn von v vermöge einer nicht-matching Kante mit „v“ markiert, zu jedem nicht von M überdeckten Punkt $v \in W$ die Nachbarn vermöge einer matching Kante. So kann beginnend bei einem Punkt in W_1, der bei der Markierung (Schritte (1)-(2)) vor Schritt (3) gefunden wird, ein M-alternierender Weg P konstruiert werden, der in U_1 endet, da der Algorithmus sonst in dem entsprechenden Punkt $u \in U$ eine matching-Kante vermöge der Markierung von u zur Verlängerung von P wählen würde. P ist also M-vergrößernder Weg.

Ist M ein maximum matching, so gibt es keinen M-erweiternden Weg mehr in G. Alle markierten Punkte aus W inzidieren mit Nicht-matching Kanten,
Nun bleibt noch zu klären, dass der Algorithmus tatsächlich mit dem versprochenen maximum matching terminiert.
Um die Formalisierung zur Korrektheitsanalyse zu verstehen, wird folgende Umformulierung der Methode betrachtet.

Bemerkung 14.
Die markierten Punkte in jedem Schritt induzieren einen Untergraphen F von G, der ein Wald ist, dessen Komponenten alle mit einem mit „0“ markierten Punkt beginnen und deren Punkte in W alle Grad 2 haben. Nach jedem Erweiterungsschritt fällt eine Komponente von F weg, deren Endpunkt in W_1 lag, so dass am Ende des Verfahrens keine Verbindung zwischen W_1 und F mehr besteht.

Dies wird nun formal beschrieben.

Satz 21 (Korrektheit der Ungarischen Methode).
Sei $G = (V, E)$ ein bitpartiter Graph mit Bipartition $V = U \uplus W$ und $M \subseteq E$ ein matching in G. Ferner seien

$$\begin{aligned} U_1 &= \{u \in U \mid u \notin e \text{ für alle } e \in M\}, \\ W_1 &= \{w \in W \mid w \notin e \text{ für alle } e \in M\} \end{aligned}$$

und $F = (V, E')$ ein Wald in G, der maximal ist bezüglich folgender Bedingungen:

1. *Jede Komponente von F enthält genau einen Punkt aus U_1.*
2. $\operatorname{grad}(w) = 2$ *für jedes $w \in W \cap V$ und $w \in e$ für ein $e \in M$.*

Es gilt:

- *M ist genau dann ein maximum matching, wenn es keine Kante $uw \in E$ gibt mit $w \in W_1$ und $u \in V(F)$.*
- *Dann ist $(V(F) \cap W) \cup (U \setminus V(F))$ ein minimum vertex cover.*

Beweis:

- „$\Rightarrow$“: Sei M ein maximum matching in G. Angenommen, es existiere eine Kante $uw \in E$ mit $u \in V(F)$ und $w \in W_1$. Sei P die Komponente von F, die u enthält. Sei $u' \in U_1 \cap P$ der Punkt bezüglich *1.* und P' der Weg zwischen u' und u in F. Wegen *2.* ist P' alternierend gerader Länge. Dann ist $P \cup uw$ ein M-erweiternder Weg, womit M nicht maximum war und ein Widerspruch besteht.

  „$\Leftarrow$“: Es gebe keine Kante zwischen $V(F)$ und W_1. Angenommen, M sei nicht maximum. Dann gibt es einen M-erweiternden Weg $P = v_1 \dots v_k$ in G. Dann gibt es ein $w \in P \cap W$,

welches nicht von M überdeckt ist und somit $w \in W_1$.[23] Die anderen Punkte in $P \cap W$ haben jedoch Grad 2, womit P zu einer Komponenten von F gehört und nicht mehr mit W_1 verbunden ist. Wirderspruch.

- Sei $uw \in E$ mit $u \in U$ und $w \in W$.

  1. $uw \in M$:
     (a) Sei $w \notin V(F) \cap W$. Wäre $u \notin U \setminus V(F)$, so wäre $u \in U \cap V(F)$ und insbesondere nicht in U_1. Dann wäre aber auch w in F, Widerspruch. Somit ist $u \in U \setminus V(F)$.
     (b) Sei $u \notin U \setminus V(F)$, also $u \in V(F) \cap U$, so tritt obige Situation wieder ein.
  2. $uw \notin M$:
     (a) Sei $w \notin V(F) \cap W$. Wäre $u \in U \cap V(F)$, dann gäbe es eine Kante uw' mit $w \neq w' \in V(F) \cap W$. Sei zunächst $u \in U_1$ und über uw', $w' \neq w$ mit F verbunden. Dann ist im Widerspruch zur ersten Aussage des Satzes $w \in W_1$, da sonst wegen der Maximalität von F beziehungsweise der Markierung durch den Algorithmus bereits w innerhalb der F-Komponenten von u läge.
     Sei also $u \notin U_1$ und o.B.d.A. $uw' \in M$. In dem Fall ist wieder $w \in W_1$, weil sonst ein längerer M-alternierender Weg über u konstruierbar wäre, Widerspruch.
     Also ist $U \in U \setminus V(F)$.
     (b) Sei $u \in V(F) \cap U$. Der Fall $w \in W \setminus V(F)$ führt wie im vorangegangenen Fall zum Widerspruch.

  $(W \cap V(F)) \cup (U \setminus V(F))$ ist minimal, da F maximal mit seinen Eigenschaften ist.

□

Das Verfahren hat eine polynomielle Laufzeit: Die Markierung kann höchstens $|W|$ mal ablaufen, da höchstens $|W|$ M-erweiternde Wege mit Anfangspunkt in W_1 gefunden werden. Die Markierung selbst erfordert für jedes $u \in U$ im schlechtesten Fall die Betrachtung aller Knoten.

Bemerkung 15.
Die ungarische Methode hat eine Laufzeit in $\mathcal{O}(|U| \cdot |V| \cdot |W|)$.

Etwas komplexer aber durchaus noch gut lösbar ist die gewichtete Variante des bitpartiten matching-Problems auf vollständig bipartiten Graphen. Diese kann nach einer geeigenten Vorverarbeitung der Eingabedaten erneut mit der ungarischen Methode gelöst werden.

[23] Hier kann davon ausgegangen werden, dass P bereits M-alternierend ist für nichtleeres M, sonst wähle eine beliebige Kante zu M.

Problem 12 (WEIGHTED BIPARTITE MATCHING).

Gegeben:	*Vollständiger bipartiter Graph* $G = (U \uplus W, E)$,
	$U = \{u_1, \ldots, u_n\}$, $W = \{w_1, \ldots, w_n\}$,
	Kostenfunktion $c : E \to \mathbb{R}$.
Gesucht:	*maximum matching* M *in* G *mit* $c(M) = \sum_{e \in M} c(e)$ *minimal,*
	$C \in \mathbb{R}^{n \times n}$ *Kostenmatrix von* G *vermittels* c.

Die Lösung dieses Problems wird im folgenden informell vorgestellt:

1. Ein matching $M \subseteq E$ ist äquivalent zu einer Permutation $\pi \in S_n$ in folgendem Sinne:

   $$u_i w_j \in M :\Longleftrightarrow \pi(i) = j$$

   Somit kann die Kostenmatrix dargestellt werden als

   $$C = (c(u_i w_j))_{1 \leq i,j \leq n} = (c_{i\pi(i)})_{1 \leq i \leq n}.$$

2. Ein maximum matching mit der geforderten Minimierungseigenschaft bleibt invariant unter zeilenweiser und spaltenweiser Addition von Zahlen auf die Einträge der Kostenmatrix: Seien $X, Y \in \mathbb{R}^n$ und $\pi*$ die optimale Lösung.

   $$\sum_i c_{i\pi^*(i)} = \min_{\pi \in S_n} \sum_i c_{i\pi(i)} \Longleftrightarrow \sum_i (c_{i\pi^*(i)} + x_i + y_{\pi^*(i)}) = \min_{\pi \in S_n} \sum_i (c_{i\pi(i)} + x_i + y_{\pi(i)}).$$

3. Durch Subtraktion des Spaltenminimums je Spalte und danach des Zeilenminimums je Zeile kann C zu einer Matrix $C' \in \mathbb{R}^{n \times n}_{\geq 0}$ umgeformt werden.

4. Suche nun mit der ungarischen Methode ein maximum matching M auf $G' = (U \uplus W, E')$ mit $u_i w_j \in E' \Longleftrightarrow c(u_i w_j) = 0$.

   Falls $|M| = n$ ist, stoppe. Sonst muss C' noch weiter bearbeitet werden.

Der folgende Algorithmus findet ein maximum matching auf allgemeinen Graphen. Dazu sind einige Vorüberlegungen nötig, welche in Abschnitt 2.1.2 entwickelt werden. Im Algorithmus ist auch noch das Kontraktionslemma wichtig.

Lemma 5 (Kontraktionslemma). *Seien* M_0 *ein matching in* G *und* C *ein Kreis der Länge* $2k + 1$ *mit* k *Kanten aus* M_0.[24]
Konstruiere einen kleineren Graphen $G' = (V', E')$ *durch Kontraktion*[25] *von* C*:*

- $V' := (V \setminus V(C)) \cup \{x_0\}$*, wobei* x_0 *ein neuer Knoten* $\notin V$ *ist.*
- $E' := E(G[V \setminus V(C)]) \cup \{yx_0 \mid y \in V \setminus V(C) :$ *es ex.* $z \in V(C)$ *mit* $yz \in E\}$.[26]

[24] Also ist eine Ecke unüberdeckt.

[25] Zusammenziehen

[26] Das bedeutet, C aus dem ursprünglichen Graphen G wird zu einem Punkt x_0 kontrahiert, woraus der verkleinerte Graph G' entsteht.

- $M_0' := M_0 \setminus E(C)$.

M_0 ist genau dann ein maximum matching in G, wenn M_0' ein maximum matching in G' ist.[27]

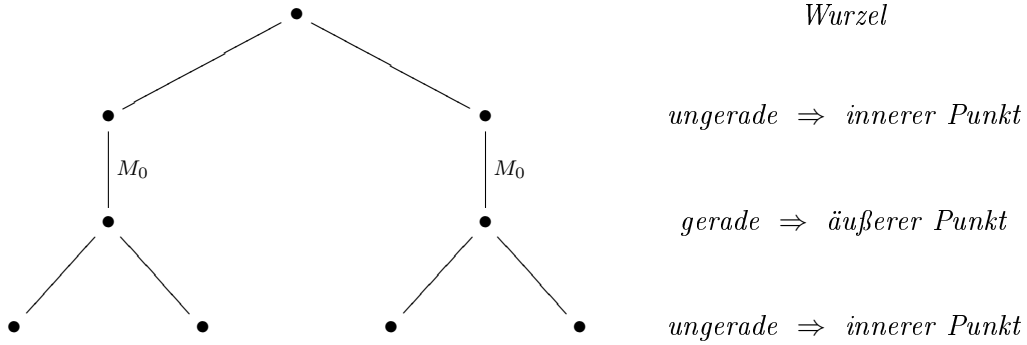
Beweis: „⇒“:
„⇐“: Das „Aufblähen“ eines beliebigen Punktes in einem Graphen □

Der Struktursatz beschreibt die Situation während der Laufzeit des EDMONDS-Algorithmus und behandelt die verschiedenen, auftretenden Spezialfälle.

Satz 22 (Struktursatz zum EDMONDS-Algorithmus).
Sei M_0 ein matching in G und $F \subseteq G$ ein maximaler Wald[28] *mit folgenden Eigenschaften:*

*(F1) Jede Komponente von F enthält genau von M_0 nicht überdeckten Punkt, die **Wurzel**.*

*Punkte mit ungeradem Abstand zur Wurzel heißen **innere Punkte**, Punkte mit geradem Abstand **äußere Punkte**.*[29] *Skizze:*



(F2) Jeder innere Punkt hat Grad 2 in F.[30]

Dann:

(1) Kein $e \in E$ verbindet einen äußeren Punkt von F mit $V \setminus V(F)$. [31]

(2) Sind zwei äußere Punkte zweier verschiedener Komponenten von F in G adjazent, so ist M_0 kein maximum matching.

(3) Sind zwei äußere Punkte in der gleichen Komponente von F adjazent, so kann der entstehende Kreis nach Kontraktionslemma kontrahiert werden und ein matching M_1 konstruiert werden mit $|M_1| = |M_0|$.

[27] Das heißt, das Kontrahieren von Kreisen beeinflusst ein maximum matching nicht.
[28] nicht im Sinne von Satz 1 sondern maximal bezüglich (F1) und (F2)
[29] Vergleiche 1. in Satz 21.
[30] Vergleiche 2. in Satz 21.
[31] Vergleiche Punkt 1 in Satz 21.

(4) Falls keine Adjazenz zwischen äußeren Punkten besteht, so ist M_0 maximum matching in G.[32]

Die Komponenten von F heißen auch ***ungarische Bäume****.*

Beweis:

(1) Der EDMONDS-Algorithmus vergrößert den Wald F mit (F1) und (F2) durch Hinzufügen von Kanten in Schritt (1).

(2) Sollte diese Aussage zutreffen, wird in Schritt (2) des Algorithmus M_0 vergrößert.

(3) Kontraktion wie in Schritt (3).

(4) Abbruchbedingung des Algorithmus.

□

Der EDMONDS-Algorithmus verwendet eine Verallgemeinerung des bei der ungarischen Methode verwendeten Waldes, da hier nicht auf die Bipartition $U \uplus W$ zurückgegriffen werden kann. Diese wird „künstlich" erzeugt durch die Definition von *inneren* und *äußeren* Punkten. Die ungarische Methode jedoch ist auf bipartiten Graphen effizienter als der EDMONDS-Algorithmus, da dieser die speziellen Eigenschaften der 2-Färbung nicht ausnutzt und auf ganz allgemein konstruierten Bäumen arbeitet.

Bemerkung 16.
Ist $G = (U \uplus W, E)$ ein bipartiter Graph, so sind die Komponenten des Waldes F der ungarischen Methode ungarische Bäume.

Beweis: Wähle die Punkte aus U_1 als Wurzeln und die inneren Punkte aus W. □

[32] Dies ist die Abbruchbeingung für den EDMONDS-Algorithmus.

Algorithmus 4 (EDMONDS Algorithmus).

Eingabe:	*$G = (V, E)$, initiales matching $M_0 \subseteq E$,*
	Wald F mit (F1), (F2) und $V(F) = V \setminus \bigcup M_0$ nicht notwendig maximal.
(1)	Vergrößerung von F
	Für alle $xy \in E$ mit x äußerer Punkt und $y \in V \setminus V(F)$:
	Wähle Punkt $z \in V$ mit $yz \in M_0$ und setze $E(F) := E(F) \cup \{xy, yz\}$.
(2)	Vergrößerung von M_0
	Für alle $xy \in E$, so dass x, y äußere Punkte unterschiedlicher Komponenten:
	Seien W_x, W_y die Wege von den Wurzeln der jew. Komponenten zu x, y.
	Setze $M_0 := M_0 \triangle E(W_x \cup W_y \cup \{xy\})$.
(3)	Kontraktion
	Für alle $xy \in E$, so dass x, y äußere Punkte der gleichen Komponente:
	Sei z der gemeinsame Vorgänger von x und y in F.
	Falls z Wurzel ist, setze $M_1 := M_0$.
	Sonst sei P der Weg von z zur Wurzel. Setze $M_1 := M_0 \triangle E(P)$.
	Wende Kontraktionslemma auf den Kreis zwischen x, y, z und M_1 an.
(4)	*Falls keine zwei äußeren Punkte verbunden sind, stoppe.*
Ausgabe:	*maximum matching: M_0*

3.3 Flussprobleme und Anwendungen

Problem 13 (MAX FLOW).

Gegeben:	*Netzwerk (D, s, t, c) über $D = (V, A)$.*
Gesucht:	*Maximaler Fluss $x : A \to \mathbb{R}_{\geq 0}$.*

Im folgenden FORD & FULKERSON-Algorithmus wird der Fluss entlang vergrößernder s-t-Wege so lange um die minimale Restkapazität erhöht, bis es keinen solchen vergrößernden Weg mehr gibt.

Algorithmus 5 (FORD & FULKERSON-Algorithmus).

Eingabe:	*Netzwerk (D, s, t, c) über $D = (V, A)$.*
(0)	*Setze $x = 0$ Nullfluss.*
(1)	*Suche vergrößernden s-t-Weg $v_0 a_1 \dots a_k v_k$.*
	Für jedes $i = 1, \dots, k$ setze $\varepsilon_i := \begin{cases} c(a_i) - x(a_i) & a_i \text{ Vorwärtskante} \\ x(a_i) & a_i \text{ Rückwärtskante} \end{cases}$
	und wähle $\varepsilon := \min_i \varepsilon_i$.
	Setze $x'(a) := \begin{cases} x(a) & a \notin \{a_1, \dots, a_k\} \\ x(a) + \varepsilon & a \text{ Vorwärtskante} \\ x(a) - \varepsilon & a \text{ Rückwärtskante} \\ x(a) + \varepsilon & a = (t, s) \end{cases}$ und $x := x'$.
Ausgabe:	*maximaler Fluss x.*

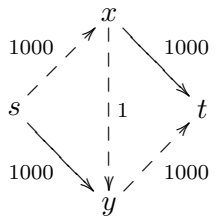
Die Leistung von Ford & Fulkerson ist der nach ihnen benannte folgende Satz über die Korrektheit ihres Algorithmus.

Satz 23 (Korrektheit des Ford & Fulkerson-Algorithmus).
Sei x ein Fluss.
x ist maximal in N genau dann, wenn kein vergrößernder s-t-Weg in N existiert.

Beweis: „⇒“: Durch Widerspruch. Unter der Annahme, dass ein vergrößernder s-t-Weg existiert, kann mittels Ford & Fulkerson-Algorithmus auch ein größerer Fluss gefunden werden.
„⇐“: Betrachte Schnitt $C = \{v \in V \mid v = s \text{ oder } v \text{ es ex. ein vergrößernder } s\text{-}v\text{-Weg}\}$ und zeige, dass dieser kapazitäts-minimal ist. Dann folgt, dass der Fluss wert-maximal ist.
Gernauer: Sei $(y, z) \in A$ eine Vorwärtskante mit $y \in C$ und $z \in V \setminus C$. Ein vergrößernder (s, y)-Weg kann zu einem vergrößernden $(s.z)$-Weg verlängert werden, falls $x(y, z) < c(y, z)$ und somit auch $z \in C$ wäre. Für Rückwärtskanten $(z, y) \in A$ mit $z \in C$ und $y \in V \setminus C$ ist $x(z, y) = 0$, weswegen die Kapazität des Schnittes $\text{cap}(C)$, die nur die Vorwärtskanten-Kapazitäten berücksichtigt, nicht von $\text{val}(x)$ abweicht. □

Sind $c(a) \in \mathbb{Z}$ für alle $a \in A$, so ist auch $x(a) \in \mathbb{Z}$, da der Algorithmus mit einem Nullfluss beginnt, welcher nur um gesamte Kapazitäten vergrößert wird. Diese Dualitätsaussage kann durch Umformulierung auf das Flussproblem auf andere Graphprobleme übertragen werden. Folgende Kritikpunkte können an den ursprünglichen Ford & Fulkerson-Algorithmus gestellt werden, so dass klar wird, dass die Laufzeit des Algorithmus zunächst unbeschränkt ist.

1. Betrachte das folgende Netzwerk (Kantenbeschriftung entspricht den Kapazitäten):

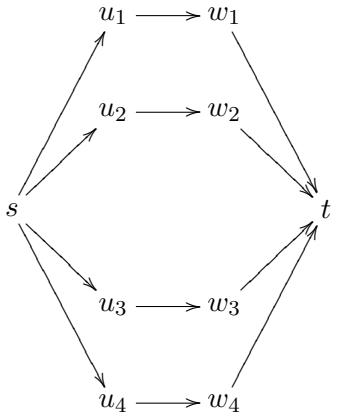


   Im Ford & Fulkerson-Algorithmus kann beispielsweise der s-t-vergrößernde Weg über die Punkte x und y verwendet werden (gestrichelt dagestellt). Der Wert des Flusses kann in dem Fall immer nur um 1 erhöht werden. Bis schließlich der maximale Wert 1000 erreicht ist, werden 1000 Schritte benötigt. Bei der sofortigen Wahl des Weges nur über x wird aber zum Beispiel nur ein Schritt benötigt.

2. Bei folgendem Netzwerk werden die Kapazitäten über der irrationale Zahl $r = \frac{-1+\sqrt{5}}{2}$[33]

[33] vergleiche goldener Schnitt

definiert: $c(a) := \frac{1}{1-r} = -\frac{2}{\sqrt{5}-3}$ für alle $a \in A$.



Der maximale Fluss x ist gegeben durch $\text{val}(x) = \frac{4}{1-r}$, der FORD & FULKERSON-Algorithmus jedoch terminiert nicht.

Um oben angesprochene Probleme zu beheben, wird die Suche nach vergrößernden s-t-Wegen des FORD & FULKERSON-Algorithmus durch eine Breitensuche ersetzt, das bedeutet, es werden nur kürzeste s-t-Wege betrachtet. So kann sogar eine obere Laufzeitschranke angegeben werden.

Satz 24 (Karp, Edmonds).
Falls in Schritt 2 des FORD & FULKERSON*-Algorithmus vergrößernde s-t-Wege mit minimaler Kantenzahl gewählt werden, so terminiert das Verfahren nach höchstens $\frac{|A||V|}{2}$ Schritten.*

Beweis:

□

Anwendungen

1. *Bipartites matching*

   Gegeben: Ein bipartiter Graph $G = (U \uplus W, E)$.

   Konstruktion: Konstruiere Netzwerk über $D = (V, E')$ für Anwendung des Algorithmus.

   - $V := U \cup W \cup \{s, t\}$.
   - $E' := \{(u, w) \mid uw \in E\} \cup \{(s, u) \mid u \in U\} \cup \{(w, t) \mid w \in W\}$.
   - Kapazitäten $c((u, w)) := r \geq 1$, $c((s, u)) := 1$, $c((w, t)) := 1$, für $w \in W$, $u \in U$.

   Ergebnis: • Nach Anwenden des FORD & FULKERSON-Algorithmus maximaler Fluss $x \in \{0, 1\}^{|A|}$ und minimaler Schnitt

   $$C = \{v \in V \mid v = s \text{ oder ex. vergrößernder } (s - v)\text{-Weg}\}$$

Dann: $M = \{e \in E \mid e = uw \text{ und } x((u,w)) = 1\}$ ist maximum matching und $Z = (U \setminus C) \cup (W \cap C)$ ein minimum vertex cover. *Bemerkungen:*

(a) M ist ein matching, da immer nur genau eine Kante mit Startpunkt $u \in U$ den Wert 1 erhält.

(b) $|M| = \text{val}(x)$ ist maximal.

(c) C hat die gleichen Eigenschaften bezüglich der Konstruktion des minimum vertex covers wie $V(F)$ aus der ungarischen Methode.
Behauptung: C stimmt genau mit den markierten Punkten $V(F)$ nach Anwenden der ungarischen Methode überein.

(d) $|Z| = \text{cap}(C)$ ist minimal.

$\Longrightarrow$ **Alternativer Beweis zum Satz von König.**

2. *Satz von* Gale *und* Ryser

Gegeben: $\lambda \in \mathbb{Z}^r$, $\mu \in \mathbb{Z}^q$, $\lambda_i \mu_i \geq 0$

Frage: Existiert eine Matrix $X \in \{0,1\}^{r \times q}$ mit $(x_{ij})_{\substack{\leq i \leq r \\ 1 \leq j \leq q}}$, so dass die Zeilen Zahlpartitionen der λ_i und die Spalten Zahlpartitionen der μ_j bilden, das heißt, für alle $i \in \{1, \ldots, r\}$ und $j \in \{1, \ldots, q\}$ gilt

$$\lambda_i = \sum_{j=1}^{q} x_{ij} \quad \text{und} \quad \mu_j = \sum_{i=1}^{r} x_{ij}.$$

Ergebnis:

3. *Satz von* Dilworth

- Ein Paar $(P, \prec)$ heißt **Halbordnung**, falls $\prec \subseteq P \times P$ transitiv und irreflexiv ist.
- $x_1 \prec \cdots \prec x_k$ heißt **Kette der Länge k in P.**
- Eine Menge paarweise unvergleichbarer Elemente heißt **Antikette in P.**
- $\mathcal{K} = \{K_1, \ldots, K_r \mid K_i \text{ Kette in } P, K_i \cap K_j = \emptyset \text{ für } i \neq j\}$ heißt **Kettenzerlegung von P.**

Die Konstruktion eines Netzwerks, auf welchem der Ford & Fulkerson-Algorithmus angewandt wird, wird im Beweis des Satzes von Dilworth gebraucht.

Gegeben: $(P, \prec)$ Halbordnung.

Satz: $\min\{|\mathcal{K}| \mid \mathcal{K} \text{ Kettenzerlegung von } P\} = \max\{|A| \mid A \text{ Antikette in } P\}$.

„$\leq$“: Eine Antikette kann mindestens so groß sein, wie die minimale Anzahl der Kettenzerlegungungen von P, da die Antikette, die aus jeweils einem Element einer der Ketten der minimalen Zerlegung besteht.

„≥“: **Konstruktion:** Umformulierung in ein bipartites matching Problem.

Definiere $G = (V := P \uplus P', E)$ mit $pq \in E \iff p \prec q$. Sei M maximum matching in G und W das duale minimum vertex cover, also $|M| = |W|$.

(a) Kettenzerlegung: Benötigen noch Kanten, die die identischen Elemente p und p' verbindet.
Setze $E \not\supseteq N := \{pp' \mid p \in P\}$, $G' := (V, N \uplus M)$. Die Komponenten von G' sind M-alternierende Wege, die Punkte aus P bilden je Komponente eine Kette in $(P, \prec)$, womit die Zerlegung von G' in Komponenten mit einer Kettenzerlegung von $(P, \prec)$ einhergeht. Da M maximum ist, ist auch die Komponenten-Zerlegung und somit die Kettenzerlegung maximal.
G' hat keine Kreise, da sonst wegen der Transitivität der Halbordnung identische Elemente p, p' vergleichbar wären und somit ein Widerspruch zur Irreflexivität bestünde. Somit hat eine Komponente von G' immer eine matching Kante weniger als die Komponente Punkte in P hat. Somit ist $|M|$+#Komponenten= $|M| + $#Ketten= $|P|$.

(b) Antikette: $A := \{p \in P \mid p \notin W \wedge p' \notin W\}$
Wäre nämlich $p \in A$ und $p \in W$, so gibt es ein $q' \in P'$ mit $p \prec q'$. Damit ist auch $q \in A$, also ist A keine Antikette (analog mit p'). Das Komplement von A ist die Projektion $W_0 := \{p \in P \mid p \in W \vee p' \in W\}$ von W auf P und somit $|W_0| \leq |W|$. Daraus folgt, dass $|P \setminus W| = |P| - |W| \leq |P| - |W'| = |P \setminus W'| = |A|$

Insgesamt gilt für eine Kettenzerlegung $\mathcal{K}$ und eine Antikette A

$$|A| \overset{2.}{\geq} |P| - |W| \overset{\text{Dualitätssatz}}{=} |P| - |M| = |\mathcal{K}|.$$

4. *Satz von* Menger

Gegeben: $D = (V, A)$, $s, t \in V$.

Satz: (a) **Gerichtete Kantenversion:** Die minimale Anzahl Kanten eines $F \subseteq A$, so dass in $(V, A \setminus F)$ kein gerichteter s-t-Weg mehr existiert ist gleich der maximalen Anzahl kantendisjunkter s-t-Wege.

(b) **Gerichtete Knotenversion:** Die minimale Anzahl Knoten eines $W \subseteq V \setminus \{s, t\}$, so dass in $(V \setminus W, A \setminus (W^2 \cup \{s, t\}))$ kein gerichteter s-t-Weg mehr existiert ist gleich der maximalen Anzahl knotendisjunkter Wege.

Die ungerichtete Version für einen Graphen (V, E) ist wie die gerichtete Version notiert, nur dass die minimale Kanten- respektive Knotenmenge nicht die gerichteten s-t Wege sondern den Zusammenhang selbst zerstören soll.

Beweis: Beweise nur die gerichtete Version, da ein ungerichteter Graph äquivalent in einen gerichteten umgewandelt werden kann durch $xy \in E \iff (x, y), (y, x) \in A$.

Kantenversion: Setze Kapazitäten $c(a) := 1$ für alle $a \in A$. Dann liefert der FORD & FULKERSON-Algorithmus einen maximalen Fluss x und einen minimalen Schnitt C, so dass $\text{val}(x)$ kantendisjunkte s-t-Wege existieren und $\text{cap}(C) = \text{val}(x)$ s-t-trennende Kanten.

Knotenversion: Simuliere die Kantenversion durch eine neue Kante je Knoten $v \in V \iff a_v \in A$.

3.4 Lineare Optimierungsprobleme

Das hier behandelte Problem ist die Suche nach deinem kosten-optimalen Fluss in einem Netzwerk. Seien dazu $D = (V, A)$ der zu Grunde liegende gerichtete Graph mit Inzidenzmatrix $B \in \{0, \pm 1\}^{n \times m}$, $m = |A|$, $n = |V|$, $b \in \mathbb{R}^m$ der *Bedarfsvektor*, $u \in \mathbb{R}^m$ der Kapazitätsvektor des Netzwerks und c der Vektor der Kosten für jeden Bogen $a \in A$. Der Bedarf in einem Punkt $v \in V$ ist die Differenz aus dem Fluss der ausgehenden Bögen und dem der eingehenden. Ist der Bedarf in v positiv (mehr ausgehender Fluss als eingehender), so wird v als *Produktionsknoten* bezeichnet, ansonsten als *Verbrauchsknoten* bezeichnet.

Problem 14 (MIN COST MAX FLOW).

$$\begin{aligned} \min \quad & c^T x \\ Bx &= b \\ 0 \leq \quad x &\leq u \end{aligned}$$

Zur Berechnung des optimalen Flusses kann zum einen die complementary slackness Bedingung verwendet werden, es gibt aber noch andere Möglichkeiten. Diese werden in folgendem Algorithmus zusammengefasst.

Algorithmus 6 (Berechnung des MCF).

1. Bestimme einen zulässigen Fluss x mit dem FORD & FULKERSON-Algorithmus.
   (a) *Erweitere D um Quelle s und Senke t, so dass für jeden Punkt $v \in V$ ein Bogen (s, v) mit Kapazität b_v (Bedarf) und ein Bogen (v, t) mit Kapazität $-b_v$ existiert.*[34]
   (b) *Suche Fluss x in erweitertem Netwerk, so dass* $\text{val}(x) = \sum_{v \in V} b_v$.[35]
2. Konstruiere Residualnetzwerk $D(x)$.

   *Ersetze jedes $(v, w) \in A$ durch sowohl (v, w) mit Kapazität $r_{(v,w)} := u_{(v,w)} x_{(v,w)}$ und Kosten $c_{(v,w)}$ als auch (w, v) mit Kapazität $-r_{(w,v)} := x_{(v,w)}$ und Kosten $-c_{(v,w)}$. $r_{(v,w)}$ bezeichnet die **Restkapazität** der Vorwärtskante (v, w). Jeder Bogen mit Restkapazität 0 wird stets aus $D(x)$ entfernt.*

[34] Das heißt, s ist einziger Produktionsknoten, t einziger Verbrauchsknoten.
[35] Das heißt, x deckt genau den Bedarf b ab.

3. Bestimme optimalen Fluss mittels CC-Algorithmus auf $D(x)$.

(a) Suche Kreis C mit negativen Gesamtkosten in $D(x)$:

i. Bestimme $\lambda =: \frac{1}{|C|} \sum_{a \in A(C)} c_a$ nach dem Satz von Karp.[36]

ii. Subtrahiere λ von allen Kosten (auf einer Kopie von $D(x)$).

iii. Suche gerichteten Kreis mit Gesamtkosten 0, dieser ist C.

(b) Vergrößere den Fluss in C wie im Ford & Fulkerson*-Algorithmus um die minimale Restkapazität. (Dann fallen bestimmte Bögen aus C im Residualnetzwerk auf Grund von Restkapazität 0 weg, womit der Kreis entfernt ist.)*

(c) Ist kein solcher Kreis mehr auffindbar, so ist x optimal.

Der in *(*3) gewählte Kreis ist ein **minimum mean circle** (**mmc**). Der Satz von **Karp** bestimmt auf konstruktive Weise die minimalen Gesamtkosten dieses Kreises im Residualnetzwerk.

Zur Termination Algorithmus 6 terminiert, da der CC-Algorithmus im letzten Schritt terminiert. Dies gilt intuitiv, da

1. nach jedem Schritt ein Kreis aus dem Residualnetzwerk entfernt wird (durch Entfernen ein oder mehrerer Bögen mit Restkapazität 0),
2. das in jedem Schritt berechnete λ stets größer wird.

3.5 Approximation

Die NP-vollständige Version des Problem des Handlungsreisenden (Travelling Salesman Problem) ist Ausgangspunkt dieses Beispiels. Dies wird hier zunächast als Entscheidungsproblem formuliert.

Problem 15 (TSP).

Gegeben:	*Graph $G = (V, E)$ auf* Städten $V = \{s_1, \ldots, s_n\}$*, Distanz $d : V \times V \to \mathbb{Z}_{\geq 0}$, Schranke $B \leq \sum_{i,j} d(s_i, s_j)$*
Frage:	*Gibt es ein $\pi \in S_n$ mit $\sum_{i=1}^{n-1} d(s_{\pi(i)}, s_{\pi(i+1)}) + d(s_n, s_1) \leq B$?*

In der Optimierung ist eine Rundreise $R \subseteq V^k$ für ein $k \in \mathbb{N}$ mit $\sum_{i=1}^{k} -1 d(v_{i-1}, v_i)$ minimal gesucht. Diese kann wie folgt ermittelt werden:

1.

[36] Vergleiche: Satz von Edmonds und Karp zur Verbesserung des Ford & Fulkerson-Algorithmus.

Für die Distanz d sind im Allgemeinen beliebige Werte zugelassen. Wähle beispielsweise $G = (\{v_1, \dots v_n\}, E)$ und

$$d(v_i, v_j) := \begin{cases} k, & s_i s_j \in E \\ 1 & \text{sonst} \end{cases} \qquad k \gg n, i \neq j.$$

In dem Fall erfüllt d nicht die Dreiecksungleichung ($\triangle$-Ungleichung). Seien nämlich $s_1 s_3 \in E$ und $s_2 s_3,\ s_1 s_2 \notin E$, dann gilt

$$k = d(s_1, s_3) \not\leq d(s_1, s_2) + d(s_2, s_3) = 1 + 1 = 2.$$

Ansatz: Fordere die Gültigkeit der $\triangle$-Ungleichung für die Distanz der Eingabe des TSP.

Problem 16 (TSP$\triangle$).

Gegeben:	*Graph $G = (V, E)$, Distanz $d : V \times V \to \mathbb{R}$ mit $\triangle$-Ungleichung.*
Gesucht:	*Rundreise $R \subseteq V^k$ für ein $k \in \mathbb{N}$ mit $\sum_{i=1}^{k} -1 d(v_{i-1}, v_i)$ minimal.*

Dieses Problem löst der folgende Algorithmus mit $\varepsilon = \frac{1}{2}$.
Da nicht in jedem Graphen eine Euler-Tour existiert, müssen in einer Rundreise Kanten mehrfach benutzt werden, was in einem Multigraphen resultiert. Dieser wird in den Schritten (2) und (3) des Algorithmus konstruiert.

Algorithmus 7 (Algorithmus von Christofides).

Eingabe:	*Graph $G = (V, E)$, Distanz $d : V \times V \to \mathbb{R}$ mit $\triangle$-Ungleichung.*
(1)	*Bestimme MST T von G vermöge der*
	Kostenfunktion $c : E \to \mathbb{R} : xy \mapsto d(x, y)$.
(2)	$W := \{v \in V(T) \mid \ 2 \mid \deg v\}$.
	Bestimme kosten-minimales matching M von $G[W]$ mit $\frac{\lvert W \rvert}{2}$ Kanten.
(3)	*Füge die Kanten von M zu T als disjunkte Kopien hinzu,*
	so dass ein Multigraph H entsteht.
	H ist Eulersch, da alle ungeraden Punkte nun gerade sind.
(4)	*Setze $E := v_1 \dots v_k$ mit $v_i \in V$ Euler-Tour.*
	Streiche alle doppelten Vorkommen von Punkten und erhalte E'.
Ausgabe:	*minimale Rundreise E'*

Satz 25 (Korrektheit des Algorithmus von Christofides).
Die Aussgabe E' ist eine minimale Rundreise bezüglich der Distanz d.

Beweis: E' ist eine Rundreise mit minimalen Kosten.
E ist eine Rundreise. T besitzt nur minimale Distanzen und M ebenfalls. Streichen von Kanten verletzt wegen der $\triangle$-Ungleichung die Minimalität nicht.

□

Index