

Vorlesung  
**Kombinatorische Optimierung**  
(Wintersemester 2014/15)  
Kapitel 6: Formulierungen und Lösungskonzepte

Volker Kaibel

Otto-von-Guericke Universität Magdeburg

(Version vom 13. April 2015)

# 0/1-Knapsack Problem (Problem 5.1)

## Problem (0/1-Knapsack Problem)

Instanz: Gewichte  $g \in \mathbb{Q}_+^n$ , Gewichtsschranke  $G \in \mathbb{Q}_+$ , Werte  $w \in \mathbb{Q}_+^n$

Aufgabe: Finde  $I^* \subseteq [n]$  mit  $g(I^*) \leq G$ , so dass

$$w(I^*) = \max\{w(I) \mid I \subseteq [n], g(I) \leq G\}$$

ist.

# 0/1-Knapsack Problem (Problem 5.1)

Variablen:  $x \in \{0, 1\}^n$

Bedeutung:  $x_i = 1 \iff i \in I$

$$\begin{array}{ll} \max & \langle w, x \rangle \\ \text{s.t.} & \langle g, x \rangle \leq G \\ & x \in \{0, 1\}^n \end{array}$$

# Bin-Packing Problem (Problem 5.9)

## Problem (Bin-Packing Problem)

Instanz: Zahlen  $a_1, \dots, a_n \in \mathbb{Q}_+$

Aufgabe: Finde  $f : [n] \rightarrow [k]$  mit  $\sum_{i:f(i)=j} a_i \leq 1$  für alle  $j \in [k]$ ,  
so dass  $k$  minimal ist.

# Bin-Packing Problem (Problem 5.9)

Variablen:  $x \in \{0, 1\}^{n \times k}$

Bedeutung:  $x_{ij} = 1 \iff f(i) = j$

Variablen:  $y \in \{0, 1\}^n$

Bedeutung:  $y_j = 1 \iff$  Es gibt  $i \in [n]$  mit  $f(i) = j$

$$\begin{array}{ll} \max & \langle \mathbb{1}_n, y \rangle \\ \text{s.t.} & \langle a, x_{*,j} \rangle \leq 1 \quad \text{für alle } j \in [n] \\ & x_{ij} \leq y_j \quad \text{für alle } j \in [n] \\ & x \in \{0, 1\}^{n \times k} \\ & y \in \{0, 1\}^k \end{array}$$

# Maschinen-Scheduling (Problem 5.13)

## Problem (Makespan-Minimierung)

Instanz: Dauern  $p \in \mathbb{Q}_+^n$  von  $n$  Jobs, Anzahl  $m \in \mathbb{N}$   
gleichartiger Maschinen

Aufgabe: Aufteilung  $f : [n] \rightarrow [m]$  der Jobs auf die Maschinen,  
so dass

$$\max\left\{ \sum_{i:f(i)=j} p_i \mid j \in [m] \right\}$$

möglichst klein ist.

# Maschinen-Scheduling (Problem 5.13)

Variablen:  $x \in \{0, 1\}^{n \times m}$

Bedeutung:  $x_{ij} = 1 \iff f(i) = j$

Variablen:  $y \in \mathbb{Q}_+$

Bedeutung:  $y$  obere Schranke an  $\sum_{i:f(i)=j} p_i$  für jedes  $j \in [m]$

$$\begin{aligned} \min \quad & y \\ \text{s.t.} \quad & \sum_{j=1}^m x_{ij} = 1 && \text{für alle } i \in [n] \\ & \sum_{i=1}^n p_i x_{ij} \leq y && \text{für alle } j \in [m] \\ & x \in \{0, 1\}^{n \times m} \\ & y \in \mathbb{Q}_+ \end{aligned}$$

# Standortplanung (Problem 5.12)

## Problem (Facility Location Problem)

Instanz: *Endliche Mengen  $\{1, \dots, K\}$  von Kunden und  $\{1, \dots, S\}$  von möglichen Standorten mit Fixkosten  $f_i \in \mathbb{Q}_+$  für das Öffnen von Standort  $i$  und Service-Kosten  $c_{ij} \in \mathbb{Q}_+$  für das Bedienen von Kunde  $j$  vom eröffneten Standort  $i$  aus.*

Aufgabe: *Teilmenge  $X^* \subseteq [S]$  von zu öffnenden Standorten und Zuordnung  $\sigma : [K] \rightarrow X^*$ , so das*

$$\sum_{i \in X^*} f_i + \sum_{j \in [K]} c_{\sigma(j)j}$$

*minimal ist.*



## Standortplanung (Problem 5.12)

Variablen:  $x \in \{0, 1\}^{K \times S}$

Bedeutung:  $x_{ij} = 1 \iff \sigma(i) = j$

Variablen:  $y \in \{0, 1\}^S$

Bedeutung:  $y_j = 1 \iff$  Standort  $j$  wird eröffnet

$$\min \sum_{j=1}^S f_j x_j + \sum_{i=1}^K \sum_{j=1}^S c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^S x_{ij} = 1 \quad \text{für alle } i \in [K]$$

$$x_{ij} \leq y_j \quad \text{für alle } j \in [S]$$

$$x \in \{0, 1\}^{K \times S}$$

$$y \in \{0, 1\}^S$$

# Färbung von Graphen (Problem 5.10)

## Problem (Graphenfärbungsproblem)

Instanz: *Graph*  $G = (V, E)$

Aufgabe: *Finde eine Abbildung*  $f : V \rightarrow [k]$  mit

$$f(v) \neq f(w) \quad \text{für alle } \{v, w\} \in E,$$

*so dass  $k$  minimal ist.*

# Färbung von Graphen (Problem 5.10)

Variablen:  $x \in \{0, 1\}^{V \times [k]}$

Bedeutung:  $x_{vj} = 1 \iff f(v) = j$

Variablen:  $y \in \{0, 1\}^k$

Bedeutung:  $y_j = 1 \iff$  Farbe  $j$  wird verwendet

$$\begin{array}{ll}
 \min & \sum_{j=1}^k y_j \\
 \text{s.t.} & \sum_{j=1}^k x_{vj} = 1 \quad \text{für alle } v \in V \\
 & x_{vj} + x_{wj} \leq 1 \quad \text{für alle } j \in [k], \{v, w\} \in E \\
 & x_{vj} \leq y_j \quad \text{für alle } j \in [k] \\
 & x \in \{0, 1\}^{V \times [k]} \\
 & y \in \{0, 1\}^k
 \end{array}$$

# Maximale Schnitte (Problem 5.3)

## Problem (Maximum Cut Problem)

Instanz: *Graph*  $G = (V, E)$  mit Kantengewichten  $c \in \mathbb{Q}^E$   
Aufgabe: *Finde*  $S^* \subseteq V$  mit

$$c(\delta(S^*)) = \max\{c(\delta(S)) \mid S \subseteq V\}.$$

# Maximale Schnitte (Problem 5.3)

Vollständiger Graph:  $K_n = ([n], \binom{[n]}{2})$ .

Variablen:  $x_{v,w} \in \{0, 1\}$  (für alle  $1 \leq v < w \leq n$ )

Bedeutung:  $x_{v,w} = 1 \iff$  Kante  $\{v, w\}$  ist im Schnitt  $\delta(S)$

$$\min \sum_{v=1}^n \sum_{w=v+1}^n c_{v,w} x_{v,w}$$

$$\begin{aligned} \text{s.t.} \quad & x_{v,w} + x_{w,u} + x_{u,v} \leq 2 && \text{für alle } \{v, w, u\} \in \binom{[n]}{3} \\ & x_{v,w} - x_{w,u} - x_{u,v} \leq 0 && \text{für alle } \{v, w, u\} \in \binom{[n]}{3} \\ & x_{v,w} \in \{0, 1\} && \text{für alle } 1 \leq v < w \leq n \end{aligned}$$

# Eigenschaften dieser Modelle

- ▶ Lineare Zielfunktion
- ▶ Lineare Nebenbedingungen
- ▶ Ganzzahligkeitsbedingungen an Variablen (sogar: 0/1-Variablen)
- ▶ Polynomial viele Variablen und Nebenbedingungen (**kompakte Formulierungen**)

## Problem des Handlungsreisenden (Problem 5.2)

### Problem (Traveling Salesman Problem (TSP))

Instanz: *Kantenlängen*  $c \in \mathbb{Q}^{E_n}$  des *vollständigen Graphen*  
 $K_n = (V_n, E_n)$  auf  $n$  *Knoten*

Aufgabe: *Finde Hamilton-Kreis*  $H^* \subseteq E_n$  mit

$$c(H^*) = \min\{c(H) \mid H \subseteq E_n \text{ Hamilton-Kreis}\}.$$

# Problem des Handlungsreisenden (Problem 5.2)

Variablen:  $x \in \{0, 1\}^{E_n}$

Bedeutung:  $x_e = 1 \iff e \in H$

$$\begin{array}{ll}
 \min & \langle c, x \rangle \\
 \text{s.t.} & \sum_{e \in \delta(v)} x_e = 2 \quad \text{für alle } v \in V \\
 & \sum_{e \in \delta(S)} x_e \geq 2 \quad \text{für alle } \emptyset \neq S \subsetneq V_n \\
 & x \in \{0, 1\}^{E_n}
 \end{array}$$



## Bemerkungen zum TSP-Modell

- ▶ Die Gleichungen heißen **Grad-Gleichungen**.
- ▶ Die Ungleichungen in diesem Modell heißen **Subtour-Eliminations Bedingungen**; sie garantieren, dass  $x$  charakteristischer Vektor einer zusammenhängenden Menge von Kanten ist.
- ▶ Das Modell hat exponentiell viele Subtour-Eliminations Bedingungen, für die aber das Separationsproblem effizient gelöst werden kann (MinCut Problem mit wegen  $x \geq 0$  nicht-negativen Gewichten, vgl. Separationsalgorithmus für Matchingprobleme).

# Ausfallsichere Netzwerke (Problem 5.11)

## Problem (Survivable Network Design Problem)

Instanz: Graph  $G = (V, E)$ , Kantenkosten  $c \in \mathbb{Q}_+^E$ ,  
Zusammenhangsforderungen  $r_{s,t} \in \mathbb{N}$  für alle  $s, t \in V$

Aufgabe: Finde eine Kantenteilmenge  $F^* \subseteq E$ , für die in  $G[F]$   
für jedes Paar  $s, t \in V$  wenigstens  $r_{s,t}$  paarweise  
kantendisjunkte  $s$ - $t$ -Wege existieren, so dass  $c(F^*)$   
minimal ist.

## Ausfallsichere Netzwerke (Problem 5.11)

Variablen:  $x \in \{0, 1\}^{E_n}$

Bedeutung:  $x_e = 1 \iff e \in H$

$$\begin{array}{ll} \min & \langle c, x \rangle \\ \text{s.t.} & \sum_{e \in \delta(S)} x_e \geq r_{s,t} \quad \text{für alle } s, t \in V, S \subset V, s \in S, t \notin S \\ & x \in \{0, 1\}^{E_n} \end{array}$$

Bemerkungen:

- ▶ Die Ungleichungen in diesem Modell garantieren wegen Satz 2.21 (ungerichtete Version des Satzes von Menger) die geforderten Ausfallsicherheiten.
- ▶ Auch hier kann das Separationsproblem ( $s$ - $t$  MinCut Problem) effizient gelöst werden.

# Quadratische 0/1-Optimierung (Problem 5.4)

## Problem (Quadratische 0/1-Optimierung)

Instanz: *Kostenmatrix*  $(c_{ij}) \in \mathbb{Q}^{n \times n}$

Aufgabe: *Finde 0/1-Vektor*  $x^* \in \{0, 1\}^n$  mit

$$\sum_{i,j=1}^n c_{ij} x_i^* x_j^* = \min \left\{ \sum_{i,j=1}^n c_{ij} x_i x_j \mid x \in \{0, 1\}^n \right\}.$$

# Quadratisches Zuordnungsproblem (Problem 5.5)

## Problem (Quadratic Assignment Problem (QAP))

Instanz: Zielfunktionskoeffizienten  $d_{ijkl} \in \mathbb{Q}$  ( $i, j, k, l \in [n]$ )

Aufgabe: Finde  $n \times n$  Permutationsmatrix  $(x_{ij}^*) \in \{0, 1\}^{n \times n}$  mit

$$\sum_{i,j,k,l=1}^n d_{ijkl} x_{ij}^* x_{kl}^*$$
$$= \min \left\{ \sum_{i,j,k,l=1}^n d_{ijkl} x_{ij} x_{kl} \mid (x_{ij})^{n \times n} \text{ Permut.-Matrix} \right\}.$$

# Eigenschaften dieser Modelle

- ▶ Nicht-lineare Zielfunktion (i.a. noch nicht einmal konvex)
- ▶ Lineare Nebenbedingungen
- ▶ 0/1-Variablen
- ▶ Ein Ansatz: Linearisierung mittels folgenden Lemmas.

## Lemma 6.1

Für  $a, b, c \in \{0, 1\}$  gilt

$$c = ab \iff (c \leq a \text{ und } c \leq b \text{ und } c \geq a + b - 1).$$

# Lineares Modell für Quadratische 0/1-Optimierung

Variablen:  $x \in \{0, 1\}^n$

Variablen:  $y \in \{0, 1\}^{n \times n}$

Bedeutung:  $y_{ij} = x_i x_j$  (für  $x \in \{0, 1\}^n$ )

$$\begin{array}{ll}
 \min & \sum_{i,j=1}^n y_{ij} \\
 \text{s.t.} & y_{ij} - x_i \leq 0 \quad \text{für alle } i \in [n] \\
 & x_i + x_j - y_{ij} \leq 1 \quad \text{für alle } i \in [n] \\
 & x \in \{0, 1\}^n
 \end{array}$$

# Lineares Modell für Quadratisches Zuordnungsproblem

Variablen:  $x_{ij} \in \{0, 1\}^{n \times n}$

Variablen:  $y \in \{0, 1\}^{[n]^4}$

Bedeutung:  $y_{ijkl} = x_{ij}x_{kl}$  (für  $x \in \{0, 1\}^{n \times n}$ )

$$\begin{array}{ll}
 \min & \sum_{i,j,k,l=1}^n y_{ijkl} \\
 \text{s.t.} & \sum_{j=1}^n x_{ij} = 1 \quad \text{für alle } i \in [n] \\
 & \sum_{i=1}^n x_{ij} = 1 \quad \text{für alle } j \in [n] \\
 & y_{ijkl} - x_{ij} \leq 0 \quad \text{für alle } i, j \in [n] \\
 & x_{ij} + x_{kl} - y_{ijkl} \leq 1 \quad \text{für alle } i, j, k, l \in [n] \\
 & x \in \{0, 1\}^{n \times n} \\
 & y \in \{0, 1\}^{[n]^4}
 \end{array}$$



# Bemerkungen

- ▶ Die obigen ganzzahligen linearen Optimierungsmodelle kann man prinzipiell mit Standardverfahren für die ganzzahlige lineare Optimierung berechnen (siehe VL im SoSem 15).
- ▶ Für stark strukturierte kombinatorische Optimierungsprobleme wie die obigen gibt es jedoch in der Regel wesentlich bessere spezialisierte Verfahren.
- ▶ Die wichtigsten Konzepte sind hier *Branch-and-Bound Algorithmen* (vor allem unter praktischen Gesichtspunkten) und *Approximationsalgorithmen* (vor allem unter theoretischen Gesichtspunkten).
- ▶ Auch ganzzahlige nicht-lineare (in der Regel konvexe) Modelle spielen in der kombinatorischen Optimierung eine wichtige Rolle; wir werden solche Modelle z.B. für das Max-Cut Problem untersuchen.

# Gerüst des Branch-and-Bound Verfahrens

Situation:

- ▶ Endliche Menge  $\mathcal{X}$  von zulässigen Lösungen (die in der Regel nicht explizit vorliegen).
- ▶ Für jedes  $X \in \mathcal{X}$  sei  $c(X) \in \mathbb{Q}$  der Zielfunktionswert von  $X$ .
- ▶ Ziel: Finde  $X^* \in \mathcal{X}$  mit  $c(X^*) = \min\{c(X) \mid X \in \mathcal{X}\}$ .

Algorithmus:

- 1: Bestimme eine **primale Schranke**  $U \in \mathbb{Q} \cup \{\infty\}$  mit

$$\min\{c(X) \mid X \in \mathcal{X}\} \leq U$$

und, falls  $U \neq \infty$ , ein  $X^{\text{best}} \in \mathcal{X}$  mit  $c(X^{\text{best}}) = U$ .

- 2: Initialisiere den Branch-and-Bound Baum mit seinem **Wurzelknoten**  $\mathcal{X}$ .
- 3: Initialisiere die Menge  $A = \{\mathcal{X}\}$  der **aktiven Branch-and-Bound Knoten**.

- 4: **while**  $A \neq \emptyset$  **do**
- 5:     Wähle einen aktiven Knoten  $S \in A$ .
- 6:      $A \leftarrow A \setminus \{S\}$
- 7:     **if**  $|S| = 1$  **then**
- 8:         Sei  $S = \{X\}$
- 9:         **if**  $c(X) < U$  **then**
- 10:              $X^{\text{best}} \leftarrow X$
- 11:              $U \leftarrow c(X)$
- 12:             Gehe zu Schritt 4.
- 13:     (*Bound:*) Bestimme eine **duale Schranke**  $L \in \mathbb{Q}$  mit
- $$L \leq c(X) \quad \text{für alle } X \in S$$
- für das **Subproblem**  $S$ .
- 14:     **if**  $L < U$  **then**
- 15:         (*Branch:*) Überdecke  $S = S_1 \cup \dots \cup S_r$  mit  $r \geq 2$  (oft paarweise disjunkten) Subproblemen  $S_1, \dots, S_r \neq \emptyset$ .
- 16:         Erweitere  $B$  durch Anhängen der  $r$  neuen Branch-and-Bound Knoten  $S_1, \dots, S_r$  an  $S$ .
- 17:      $A \leftarrow A \cup \{S_1, \dots, S_r\}$

# Bemerkungen zum B&B-Verfahren

- ▶ Das Verfahren löst das Minimierungsproblem.
- ▶ Bricht man das Verfahren vor seiner Terminierung ab, so kann man zumindest für Probleme mit nicht-negativen Zielfunktionen **a posteriori** eine beweisbare Abschätzung der Qualität der bis dahin besten gefundenen Lösung  $X^{\text{best}}$  geben:
  - ▶ Man berechnet untere Schranken  $L^S$  für alle noch aktiven Subprobleme  $S \in A$ .
  - ▶ Ist

$$L_{\min} = \min\{L^S \mid S \in A\} > 0,$$

so ist der Wert  $c(X^{\text{best}})$  von  $X^{\text{best}}$  höchstens das  $\frac{c(X^{\text{best}})}{L_{\min}}$ -fache des Optimalwerts.

- ▶ Maximierungsprobleme lassen sich analog lösen.

## Bemerkungen zum B&B-Verfahren

- ▶ In Schritt 1 kann man alle Arten von Heuristiken einsetzen (z.B. Simulated Annealing, evolutionäre Algorithmen, Tabusuche, lokale Suche, . . . ); diese sind jedoch kein Thema dieser Vorlesung. (Je besser die initiale primale Schranke ist, umso öfter ist die Bedingung in Schritt 14 nicht erfüllt, und umso kleiner bleibt der Branch-and-Bound Baum.)
- ▶ In Schritt 13 kann man zusätzlich versuchen, mit Heuristiken für das Subproblem  $S$  die primale Schranke  $U$  zu verbessern.
- ▶ Wichtige Faktoren für die Effizienz:
  - ▶ Qualität der dualen Schranken
  - ▶ Effizienz der Bestimmung der dualen Schranken
  - ▶ Zusammenspiel der Branching-Regel (Art der erzeugten Subprobleme) und der Bounding-Prozedur
- ▶ Größter Teil der mathematischen Arbeit: Strukturanalyse für den Entwurf von Bounding-Prozeduren
- ▶ In der Regel hat man keine (nicht trivialen) Abschätzungen für die Laufzeit.

## LP-basiertes B&B

- ▶ Sei das (kombinatorische) Optimierungsproblem als ganzzahliges lineares Optimierungsproblem

$$\min\{\langle c, x \rangle \mid Ax \leq b, x \in \mathbb{Z}^n\}$$

formuliert ( $A \in \mathbb{Q}^{m \times n}$ ,  $b \in \mathbb{Q}^m$  so, dass  $Ax \leq b$  nur endlich viele ganzzahlige Lösungen besitzt).

- ▶ Möglichkeit für das Branching: Wähle  $i \in [n]$  und  $\kappa \in \mathbb{Z}$  (geeignet) und partitioniere

$$\mathcal{S} = \{x \in \mathbb{Z}^n \mid A^{\mathcal{S}}x \leq b^{\mathcal{S}}\}$$

in

$$\mathcal{S}_1 = \mathcal{S} \cap \{x \in \mathbb{Z} \mid x_i \leq \kappa\} = \{x \in \mathbb{Z}^n \mid A^{\mathcal{S}_1}x \leq b^{\mathcal{S}_1}\}$$

und

$$\mathcal{S}_2 = \mathcal{S} \cap \{x \in \mathbb{Z} \mid x_i \geq \kappa + 1\} = \{x \in \mathbb{Z}^n \mid A^{\mathcal{S}_2}x \leq b^{\mathcal{S}_2}\}$$

(allgemeiner: *branching-on-hyperplanes*).

## LP-basiertes B&B

- ▶ Bestimme duale Schranken durch Lösen der **LP-Relaxierungen**

$$\min\{\langle c, x \rangle \mid A^{\mathcal{S}}x \leq b^{\mathcal{S}}, x \in \mathbb{Q}^n\}$$

$$(\leq \min\{\langle c, x \rangle \mid A^{\mathcal{S}}x \leq b^{\mathcal{S}}, x \in \mathbb{Z}^n\})$$

(kontinuierliche lineare Optimierungsprobleme).

- ▶ Sei  $x^{\mathcal{S}} \in \mathbb{Q}^n$  die gefundene Optimallösung der LP-Relaxierung des aktuellen Subproblems  $\mathcal{S}$
- ▶ Falls  $x^{\mathcal{S}} \in \mathbb{Z}^n$  (also  $x^{\mathcal{S}} \in \mathcal{S}$ ):
  - ▶ Falls  $\langle c, x^{\mathcal{S}} \rangle < U$ : Setze am Ende von Schritt 13  $X^{\text{best}} \leftarrow x^{\mathcal{S}}$ .
  - ▶ Dann wird der Test in Schritt 14 in jedem Fall negativ ausfallen.
- ▶ Falls  $x^{\mathcal{S}} \notin \mathbb{Z}^n$  (und der Test in Schritt 14 positiv ausfällt): Wähle für das Branching  $i \in [n]$  so, dass  $x_i^{\mathcal{S}} \notin \mathbb{Z}$  und  $\kappa = \lfloor x_i^{\mathcal{S}} \rfloor$ .

## LP-basiertes B&B

- ▶ Die LP-Relaxierung kann auch dynamisch (mittels Schnittebenenverfahrens) gelöst werden; es genügt also, wenn das System  $Ax \leq b$  über einen Separationsalgorithmus gegeben ist (vgl. die Modelle für das TSP und das Survivable Network Design Problem in Abschnitt 6.1).
- ▶ Man kann in einem solchen Schnittebenenverfahren die dualen Schranken mit beliebigen für  $\{x \in \mathbb{Z}^n \mid Ax \leq b\}$  (oder – äquivalent dazu – für  $\text{conv}\{x \in \mathbb{Z}^n \mid Ax \leq b\}$ ) gültigen linearen Ungleichungen zu verbessern versuchen ( $\rightsquigarrow$  **Branch-and-Cut**).



## LP-basiertes B&B

- ▶ Deshalb sind möglichst gute partielle Ungleichungsbeschreibungen (mit zugehörigen Separationsalgorithmen) des der Formulierung zu Grunde liegenden Polytops

$$\text{conv}\{x \in \mathbb{Z}^n \mid Ax \leq b\}$$

sehr wichtig (vollständige Beschreibungen können wir bei komplexitätstheoretisch schwierigen Problemen ja nicht erwarten).

- ▶ Die LP-Relaxierung  $\min\{\langle c, x \rangle \mid A^S x \leq b^S, x \in \mathbb{Q}^n\}$  kann auch gelöst werden, indem man ihr duales LP mit Hilfe eines Separations-Algorithmus (**Pricing-Algorithmus**) löst ( $\rightsquigarrow$  **Column-Generation Verfahren**); so kann man in der Praxis manchmal auch ganzzahlige lineare Optimierungsmodelle mit extrem vielen Variablen lösen.

# 1-Bäume

## Definition 6.2

Ein **1-Baum** in einem Graphen  $G = ([n], E)$  ist eine Kantenteilmenge  $F \subseteq E$  mit

- ▶  $|F \cap \delta(1)| = 2$  und
  - ▶  $([n] \setminus \{1\}, F \setminus \delta(1))$  ist ein Baum (insbesondere:  $|F| = n$ ).
- 
- ▶ Hamilton-Kreise in  $G$  sind 1-Bäume in  $G$ .
  - ▶ Ist  $c \in \mathbb{R}^E$  und  $F^* \subseteq E$  ein 1-Baum minimalen  $c$ -Gewichts im Graphen  $G = ([n], E)$ , so gilt  $c(F^*) \leq c(H)$  für alle Hamilton-Kreise  $H \subset E$ .
  - ▶ Einen 1-Baum minimalen  $c$ -Gewichts kann man in  $O(|E_n|) = O(n^2)$  Zeit berechnen (minimal aufspannender Baum in  $G[[n] \setminus \{1\}]$  und die beiden  $c$ -leichtesten Kanten aus  $\delta(1)$ ); das geht auch dann noch, wenn man sich auf 1-Bäume beschränkt, die eine vorgegebene Kantenteilmenge enthalten.

## TSP: B&B mit 1-Baum-Schranke

- ▶ Branching: Wähle eine (geeignete) Kante  $e \in E_n$  und erzeuge die Subprobleme

$$\mathcal{S}_1 = \{H \in \mathcal{S} \mid e \in H\} \quad \text{und} \quad \mathcal{S}_2 = \{H \in \mathcal{S} \mid e \notin H\}.$$

- ▶ Bestimme in Schritt 13  $L$  als das Gewicht eines minimalen die Kantenmenge  $E_+^{\mathcal{S}}$  enthaltenden 1-Baums im Graphen  $([n], E_n \setminus E_-^{\mathcal{S}})$  (wobei  $E_+^{\mathcal{S}}$  bzw.  $E_-^{\mathcal{S}}$  die Mengen der Kanten sind, die im Subproblem  $\mathcal{S}$  in allen Hamilton-Kreisen enthalten bzw. nicht enthalten sind).

# Approximationsalgorithmen: Ausgangsfragen

- ▶ Kann man für NP-schwere Optimierungsprobleme polynomiale Algorithmen finden, die Lösungen mit **a priori** bekannter Gütegarantie bestimmen?
- ▶ Wie gut können diese Garantien werden?
- ▶ Unterscheiden sich einzelne NP-schwere Optimierungsprobleme hinsichtlich der prinzipiell möglichen Gütegarantien?

# Approximationsalgorithmen

- ▶ Wir betrachten hier nur kombinatorische Optimierungsprobleme, bei denen alle Lösungen *nicht-negative Zielfunktionswerte* haben.
- ▶ Ist  $k \geq 1$  und ist  $A$  ein Algorithmus, der für jede Instanz  $I$  von  $P$  in in der Kodierlänge von  $I$  **polynomial beschränkter Laufzeit** eine Lösung vom Wert  $\omega(I)$  berechnet mit

$$\omega(I) \leq k \cdot \text{OPT}(I) \quad \text{bzw.} \quad \omega(I) \geq \frac{1}{k} \cdot \text{OPT}(I)$$

– je nachdem, ob  $P$  ein Minimierungs- oder ein Maximierungsproblem ist – (wobei  $\text{OPT}(I)$  der Optimalwert der Instanz  $I$  sei), so heißt  $A$  ein  **$k$ - (Faktor-) Approximationsalgorithmus** für  $P$ .

## Beispiel: Vertex Cover Problem

Algorithmus 6.3 (2-Approximationsalgorithmus für Vertex Cover)

Eingabe: Graph  $G = (V, E)$

Ausgabe: Vertex Cover  $S \subseteq V$  von  $G$ .

- 1: Berechne ein inklusionsmaximales Matching  $M \subseteq E$  in  $G$ .
- 2:  $S \leftarrow \bigcup_{e \in M} e$

Bemerkung 6.4

Algorithmus 6.3 ist ein 2-Approximationsalgorithmus für das Vertex-Cover Problem.

# Nicht-Approximierbarkeits-Resultate: Beispiel

## Bemerkung 6.5

*Wenn es für irgendein  $k \in \mathbb{N}$  einen  $k$ -Approximationsalgorithmus für das TSP gibt, dann ist  $P = NP$ .*

- ▶ Es gibt auch gewichtete Optimierungsprobleme, für die Approximationsalgorithmen existieren (s.u.).
- ▶ Es gibt auch ungewichtete Probleme, für die für kein  $k \in \mathbb{N}$  ein  $k$ -Approximationsalgorithmus existiert, es sei denn  $P = NP$  (z.B. das stabile Mengen Problem oder das Graphenfärbungsproblem).

# Euler-Touren

## Definition 6.6

Eine **Euler-Tour** in einem Graphen  $G = (V, E)$  ist ein geschlossener Pfad  $P$ , der jede Kante aus  $E$  genau einmal benutzt.

## Lemma 6.7

*Ein Graph  $G = (V, E)$  hat genau dann eine Euler-Tour, wenn er zusammenhängend ist und alle Knotengrade  $|\delta(v)| \in 2\mathbb{Z}$  ( $v \in V$ ) gerade sind. Man kann in  $O(|E|)$  Zeit eine Euler-Tour in  $G$  bestimmen oder entscheiden, dass  $G$  keine Euler-Tour besitzt.*



# Metrische Kantenlängen

## Definition 6.8

Kantenlängen  $d \in \mathbb{R}^{E_n}$  des vollständigen Graphen  $K_n = (V_n, E_n)$  heißen **metrisch**, wenn  $d \geq \mathbb{0}_{E_n}$  ist und die Dreiecksungleichung

$$d_{\{u,w\}} \leq d_{\{u,v\}} + d_{\{v,w\}}$$

für alle paarweise verschiedenen  $u, w, v \in V_n$  gilt. Eine TSP-Instanz heißt **metrisch**, wenn ihre Kantenlängen metrisch sind.

## Bemerkung 6.9

*Ist  $P$  ein geschlossener Pfad in einem vollständigen Graphen  $K_n = (V_n, E_n)$  mit metrischen Kantenlängen  $d \in \mathbb{R}_+^{E_n}$ , der alle Knoten wenigstens einmal besucht, so kann man aus  $P$  in  $O(|P|)$  Schritten einen Hamilton-Kreis  $H \subseteq E_n$  mit  $d(H) \leq d(P)$  gewinnen.*

# Untere Schranken für metrische TSPs

## Bemerkung 6.10

*Jeder Hamilton-Kreis enthält einen aufspannenden Baum. Also gilt für für jeden Graphen  $G = (V, E)$  mit nichtnegativen Kantenlängen  $d \in \mathbb{R}^E$ , dass die  $d$ -Länge eines jeden Hamilton-Kreises in  $G$  wenigstens so groß ist wie das  $d$ -Gewicht eines  $d$ -minimalen aufspannenden Baums von  $G$*

## Lemma 6.11

*Seien  $K_n = (V_n, E_n)$  ein vollständiger Graph mit metrischen Kantenlängen  $d \in \mathbb{R}^{E_n}$  und  $W \subseteq V_n$  mit  $|V_n| \in 2\mathbb{Z}$ . Dann gilt für jedes  $d$ -minimale perfekte Matching  $M \subseteq \binom{W}{2}$  in  $K_n[W]$  und für jeden Hamilton-Kreis  $H \subseteq E_n$ :*

$$d(M) \leq \frac{d(H)}{2}$$

# Christofides Algorithmus für metrisches TSP

## Algorithmus 6.12 (Christofides Algorithmus)

Eingabe: *Metrische Kantenlängen*  $d \in \mathbb{Q}^{E_n}$  *des vollständigen Graphen*  $K_n = (V_n, E_n)$

Ausgabe: *Hamilton-Kreis*  $H \subseteq E_n$

- 1: *Bestimme*  $d$ -*minimal aufspannenden Baum*  $T \subseteq E_n$  *von*  $K_n$ .
- 2:  $W \leftarrow \{v \in V_n \mid |\delta(v) \cap T| \in 2\mathbb{Z} + 1\}$
- 3: *Bestimme*  $d$ -*minimales perfektes Matching*  $M \subseteq E_n$  *in*  $K_n[W]$ .
- 4: *Bestimme eine Euler-Tour*  $P$  *im (möglicherweise nicht einfachen) Graphen*  $(V_n, T \uplus M)$  (Lem. 6.7).
- 5: *Bestimme aus*  $P$  *einen Hamilton-Kreis*  $H \subseteq E_n$  *mit*  $d(H) \leq d(P)$  (Bem. 6.9).

## Satz 6.13

*Christofides Algorithmus ist ein  $\frac{3}{2}$ -Approximationsalgorithmus für das metrische TSP.*

## Bemerkungen zur Approximierbarkeit des TSP

- ▶ Wenn  $P \neq NP$  ist, dann gibt es nicht für jedes  $\varepsilon > 0$  einen  $(1 + \varepsilon)$ -Approximationsalgorithmus für das metrische TSP.
- ▶ Für das **Euklidische TSP** (d.h., die Knoten des Graphen entsprechen Punkten in der Euklidischen Ebene und die Kantenlängen sind die respektiven Euklidischen Distanzen) gibt es aber für jedes  $\varepsilon > 0$  einen  $(1 + \varepsilon)$ -Approximationsalgorithmus (bei dem  $\frac{1}{\varepsilon}$  allerdings als Exponent in die Laufzeitabschätzung eingeht): Das Euklidische TSP hat ein **Polynomiales Approximationsschema (PTAS)**.
- ▶ Trotzdem ist das Euklidische TSP noch NP-schwer.
- ▶ Das PTAS für Euklidisches TSP ist aber in der Praxis ausgefeilten Branch-and-Bound Algorithmen weit unterlegen.