

UNIVERSITÄT AUGSBURG

INSTITUT FÜR MATHEMATIK

Seminar „ausarbeitung“

zu einem Vortrag im Seminar Spieltheorie und Approximationsalgorithmen im SS 2016
zum Thema

Capacitated Vehicle Routing with Non-Uniform Speeds

Zusammengestellt:
Lukas GRAF

Betreut von:
M. Sc. Manuel SUREK,
Prof. Dr. Tobias HARKS

1 Problemübersicht

Metrisches TSP:

- Vollständiger Graph $G = (V, E)$
- Metr. Abstandsfunktion $d : E \rightarrow \mathbb{R}$
- **Lsgen:** Tour τ durch ganz V
- **Ziel:** Minimiere $d(\tau)$

Homogenes TSP:

- Vollständiger Graph $G = (V, E)$
- Metr. Abstandsfunktion $d : E \rightarrow \mathbb{R}$
- Startpunkt $s \in V$
- k Fahrzeuge
- **Lsgen:** Touren (τ_i) , die bei s beginnen und gemeinsam ganz V abdecken
- **Ziel:** Minimiere $\max d(\tau_i)$

Heterogenes TSP:

- Vollständiger Graph $G = (V, E)$
- Metr. Abstandsfunktion $d : E \rightarrow \mathbb{R}$
- Startpunkt $s \in V$
- k Fahrzeuge mit Geschw. $(\lambda_i)_{i=1}^k$
- **Lsgen:** Touren (τ_i) , die bei s beginnen und gemeinsam ganz V abdecken
- **Ziel:** Minimiere $\max \frac{d(\tau_i)}{\lambda_i}$

CVRP:

- Vollständiger Graph $G = (V, E)$
- Metr. Abstandsfunktion $d : E \rightarrow \mathbb{R}$
- Startpunkt/Depot $s \in V$
- Bedarfe $(q_v)_{v \in V}$ und Kapazität Q
- **Lsgen:** Tour (τ) , die bei s beginnt alle Bedarfe erfüllen nie mehr als Q Elemente transportiert
- **Ziel:** Minimiere $\max d(\tau_i)$

Homogenes CVRP:

- Vollständiger Graph $G = (V, E)$
- Metr. Abstandsfunktion $d : E \rightarrow \mathbb{R}$
- Startpunkt/Depot $s \in V$
- Bedarfe $(q_v)_{v \in V}$
- k Fahrzeuge mit einheitlicher Kapazität Q
- **Lsgen:** Touren (τ_i) , die bei s beginnen, gemeinsam alle Bedarfe erfüllen, wobei kein Fahrzeug jemals mehr als Q Elemente transportiert
- **Ziel:** Minimiere $\max d(\tau_i)$

Heterogenes CVRP:

- Vollständiger Graph $G = (V, E)$
- Metr. Abstandsfunktion $d : E \rightarrow \mathbb{R}$
- Startpunkt/Depot $s \in V$
- Bedarfe $(q_v)_{v \in V}$
- k Fahrzeuge mit Geschw. $\{\lambda_i\}$ und einheitlicher Kapazität Q
- **Lsgen:** Touren (τ_i) , die bei s beginnen, gemeinsam alle Bedarfe erfüllen, wobei kein Fahrzeug jemals mehr als Q Elemente transportiert
- **Ziel:** Minimiere $\max \frac{d(\tau_i)}{\lambda_i}$

2 Algorithmen für HomTSP und CVRP

2.1 Algorithmus für HomTSP

Algorithm 1 HomTSP-Approx

```
1: procedure HOMTSP( $G = (V, E)$ ,  $d : E \rightarrow \mathbb{R}_{\geq 0}$ ,  $k$ )
2:    $\tau \leftarrow \text{TSP-APPROX}(G, d)$ 
3:    $(\pi_i)_{i=1}^k \leftarrow$  Teile  $\tau$  durch Entfernen von Kanten in  $k$  Teilstrecken mit Länge  $\leq \frac{d(\tau)}{k}$ 
4:    $(\tau_i)_{i=1}^k \leftarrow$  Verbinde die zwei Endpunkte von  $\pi_i$  mit  $s$ .
5:   return  $(\tau_i)$ 
6: end procedure
```

Lemma 2.1 (Theorem 8 in [FHK76]¹). *Unter Verwendung eines ρ -approximativen Algorithmus (z.B. $\rho = \frac{3}{2}$ durch Christofides) für **TSP** ist HOMTSP $(\rho + 1)$ -approximativ.*

Beweis. Schritt 3 ist möglich, denn jeder Pfad π_i entspricht Kanten der Gesamtlänge $> \frac{d(\tau)}{k}$ in τ : Zunächst die (möglicherweise 0 vielen) Kanten in π_i der Gesamtlänge $\leq \frac{d(\tau)}{k}$ und dann der entfernten nächsten Kante, die die Gesamtlänge über diese Schwelle gebracht hätte. Nach dem Erstellen der ersten $(k - 1)$ Pfade sind daher höchstens noch Kanten der Länge

$$d(\tau) - \sum_{i=1}^{k-1} d(\pi_i) < d(\tau) - (k - 1) \cdot \frac{d(\tau)}{k} = \frac{d(\tau)}{k}$$

übrig, die daher von π_k abgedeckt werden können.

Da jeder der Endpunkte u und v von π_i auch in einer optimalen Lösung besucht werden muss, gilt sicher $d(s, u), d(s, v) \leq \frac{1}{2} \text{OPT}_{\text{HomTSP}}$. Außerdem ist die Vereinigung aller Touren τ_i^* einer optimalen Lösung von **HomTSP** eine zulässige Lösung von **TSP** und damit $\text{OPT}_{\text{TSP}} \leq \sum_{i=1}^k d(\tau_i^*) \leq k \cdot \max_{i=1}^k d(\tau_i^*) = k \cdot \text{OPT}_{\text{HomTSP}}$.

Zusammen ergibt dies für jede der Touren τ_i (und damit insbesondere für die längste):

$$\begin{aligned} d(\tau_i) &\leq \frac{d(\tau)}{k} + 1 \cdot \text{OPT}_{\text{HomTSP}} \leq \frac{\rho \cdot \text{OPT}_{\text{TSP}}}{k} + \text{OPT}_{\text{HomTSP}} \\ &\leq \frac{\rho \cdot k \cdot \text{OPT}_{\text{HomTSP}}}{k} + \text{OPT}_{\text{HomTSP}} = (\rho + 1) \text{OPT}_{\text{HomTSP}} \end{aligned}$$

□

¹Tatsächlich wird in [FHK76] mit einem etwas komplexeren Algorithmus sogar eine Approximationsgüte von $\rho + 1 - \frac{1}{k}$ erreicht.

2.2 Algorithmus für CVRP

Algorithm 2 CVRP-Approx

```

1: procedure CVRP( $G = (V, E)$ ,  $d : E \rightarrow \mathbb{R}_{\geq 0}$ ,  $Q$ ,  $(q_v)_{v \in V}$ )
2:    $G' \leftarrow$  vollst. Graph auf  $V' := \{v^{(j)} \mid v \in V, j = 1, \dots, q_v\}$ 
3:    $d'(v^{(i)}, u^{(j)}) := d(v, u)$ 
4:    $\tau \leftarrow \text{TSP-APPROX}(G', d')$ 
5:   for  $v \in V'$  do
6:      $(\pi_i^{(v)}) \leftarrow$  Teile  $\tau$  durch Kantenlöschen in Teilstrecken mit max.  $Q$  Knoten,
        wobei die erste Strecke bei  $v$  beginnt.
7:      $(\sigma_i^{(v)}) \leftarrow$  Verbinde die zwei Endpunkte von  $\pi_i^{(v)}$  mit  $s$ .
8:   end for
9:    $(\sigma_i) \leftarrow (\sigma_i^{(v)})$  mit  $\sum_i d'(\sigma_i^{(v)})$  minimal.
10:  return  $(\sigma_i)$ 
11: end procedure

```

Proposition 2.2 (Lemma 1 in [HK85]). *Es gilt:*

$$\text{OPT}_{\text{CVRP}} \geq \max \left\{ \text{OPT}_{\text{TSP}}, \frac{2}{Q} \sum_{v' \in V'} d'(s, v') \right\}$$

Beweis. Die erste Abschätzung ist klar (eine **CVRP**-Route ist insbesondere auch eine **TSP**-Tour).

Für die zweite Abschätzung betrachte eine optimale Route (σ_i^*) , wobei $S_i \subseteq V'$ die in der i -ten Tour besuchten Knoten seien. Dann gilt offenbar:

$$d(\sigma_i^*) \geq 2 \cdot \max \{d(s, v') \mid v' \in S_i\} \geq 2 \frac{\sum_{v' \in V'} d(s, v')}{|S_i|} \geq \frac{2}{Q} \sum_{v' \in S_i} d(s, v')$$

Da jeder Knoten aus V' in wenigstens einem S_i enthalten sein muss, folgt damit:

$$\text{OPT}_{\text{CVRP}} = \sum_i d(\sigma_i^*) \geq \frac{2}{Q} \sum_i \sum_{v' \in S_i} d(s, v') \geq \frac{2}{Q} \sum_{v' \in V'} d(s, v')$$

□

Lemma 2.3 (Lemma 2 in [HK85]²). *Unter Verwendung eines ρ -approximativen Algorithmus für **TSP** ist CVRP-APPROX $(\rho + 2)$ -approximativ.*

²Wie über eine genauere Abschätzung in [HK85] gezeigt wird, erreicht der Algorithmus sogar eine Approximationsgüte von $\rho + \lceil \frac{n'}{Q} \rceil \frac{Q-\rho}{n'}$, wobei $n' := \sum_v q_v$

Beweis. Sei $n' := \sum_v q_v = |V'|$ und $m := \lceil \frac{n'}{Q} \rceil$. Dann gilt für die Summe aller möglichen Routen:

$$\sum_{v,i} d'(\sigma_i^{(v)}) = 2m \sum_{v' \in V'} d'(s, v') + n' \cdot d(\tau)$$

Denn jeder Knoten ist genau einmal Anfangs- und einmal Endpunkt für jede der m Teilstrecken (erster Summand) und jede Route entsteht durch Weglassen von Kanten aus der Tour τ (zweiter Summand).

Die beste dieser Routen ist sicher mindestens so gut wie der Durchschnitt aus allen möglichen Routen, also:

$$\sum_i d'(\sigma_i) = 2 \frac{m}{n'} \sum_{v' \in V'} d'(s, v') + d(\tau) \leq 2 \frac{\frac{n'}{Q} + 1}{n'} \sum_{v' \in V'} d'(s, v') + d(\tau) \leq 2 \cdot \frac{2}{Q} \sum_{v' \in V'} d'(s, v') + \rho \cdot \text{OPT}_{\text{TSP}}$$

Zusammen mit Proposition 2.2 ergibt dies:

$$\sum_i d'(\sigma_i) \leq 2 \cdot \text{OPT}_{\text{CVRP}} + \rho \cdot \text{OPT}_{\text{CVRP}} \leq (2 + \rho) \cdot \text{OPT}_{\text{CVRP}}$$

□

3 Algorithmus für HetTSP

Algorithm 3 HetTSP-Approx

```

1: procedure HETSP( $G = (V, E)$ ,  $d : E \rightarrow \mathbb{R}_{\geq 0}$ )
2:   Rate  $M$  mit  $\frac{M}{2} \leq \text{OPT} \leq M$ 
3:    $\mathcal{H} := (H_i)_{i \geq 0} \leftarrow \text{LEVEL-PRIME}(G, d)$ 
      //  $\mathcal{H}$  erfüllt: Wurzel-Blatt Pfade haben aufsteigende Knoten-Level
      // und  $\forall i : \sum_{j \geq i} d(H_j) \leq 8M \sum_{j \geq i-1} 2^j \mu_j$  (wenn  $M$  korrekt geraten)
4:    $\mathcal{T} := (\mathcal{T}_i)_{i \geq 0} \leftarrow \text{DECOMPOSITION}(\mathcal{H})$ 
      //  $\mathcal{T}$  ist  $(6, 40)$ -zuweisbarer Wald
5:    $(x_{ij}) \leftarrow \text{FRACTIONALASSIGNMENT}(\mathcal{T})$ 
6:    $(\tau_i) \leftarrow \text{ROUNDINGASSIGNMENT}(x_{ij})$ 
      //  $\mathcal{T}$  ist  $(\alpha, \beta)$ -zuweisbar  $\Rightarrow (\tau_i)$  ist  $(4\alpha + 2\beta)$ -approx.
7:   return  $(\tau_i)$ 
8: end procedure

```

Satz 3.1 (Theorem 1.1 in [Gø+10]). *Algorithmus 3 ist ein $\mathcal{O}(1)$ -approximativer Algorithmus für **HetTSP**.*

Beweis. Folgende Kapitel...

□

3.1 Level-Prime

Algorithm 4 Level-Prime

```

1: procedure LEVEL-PRIME( $G = (V, E)$ ,  $d : E \rightarrow \mathbb{R}_{\geq 0}$ )
2:    $V_0 := \{v \in V \mid d(s, v) \leq M\}$ ,  $V_i := \{v \in V \mid 2^{i-1}M < d(s, v) \leq 2^i M\}$ 
3:   for  $i \geq 0$  do  $H_i \leftarrow$  Minimaler Spannbaum auf  $G[V_{\leq i}] / V_{< i}$  end for
4:   return  $(H_i)_{i \geq 0}$ 
5: end procedure

```

Lemma 3.2 (Theorem 3.3 in [Gø+10]). *Ein von Algorithmus 4 gefundener Baum $(H_l)_{l \geq 0}$ erfüllt:*

- Die Knoten-Level entlang jedes Wurzel-Blatt-Pfades sind monoton wachsend.
- $\forall j \geq 0 : \sum_{l \geq j} d(H_l) \leq 8 \cdot \text{MST}(G/V_{< j})$

Korollar 3.3 (Korollar 3.5 in [Gø+10]). *Ein von Algorithmus 4 gefundener Baum $(H_l)_{l \geq 0}$ erfüllt:*

- Die Knoten-Level entlang jedes Wurzel-Blatt-Pfades sind monoton wachsend.
- $\forall j \geq 1 : \sum_{l > j} d(H_l) \leq 8M \cdot \sum_{l \geq j} 2^l \mu_l$

3.2 Zerlegungsalgorithmus

Algorithm 5 Decomposition

```

1: procedure DECOMPOSITION( $(\mathcal{H})$ )
2:    $\mathcal{S}_0 := \{H_0\}$ ,  $\mathcal{S}_l := \text{Zerl. von } \mathcal{H} \cap E_l \text{ in Bäume mit genau einer Kante nach } V_l$ 
3:    $\mathcal{S}_l^{\geq} := \{\tau \in \mathcal{S}_l \mid d(\tau) \geq 2^{l-3}M\}$ ,  $\mathcal{S}_l^{<} := \mathcal{S}_l \setminus \mathcal{S}_l^{\geq}$ 
4:   for  $\tau \in \mathcal{S}_l^{<}$  do  $h(\tau) := \tau' \in \mathcal{S}_{l-1}^{\geq}$  mit  $\tau \cup \tau'$  zsh. (ex. eind.) end for
5:   for  $\tau \in \mathcal{S}_l^{\geq}$  do
6:      $\mathcal{T}_l(\tau) \leftarrow$  Partition von  $\tau \cup h^{-1}(\tau)$  in Bäume der Länge  $[2^{l+1}M, 2^{l+2}M]$ 
       (und evtl. ein kürzerer)
7:      $\mathcal{T}'_l(\tau) \leftarrow \{T_r \cup \{\text{kürzeste Kante zu } s\} \mid T_r \in \mathcal{T}_l(\tau)\}$ 
8:   end for
9:    $\mathcal{T}_l := \bigcup_{\tau \in \mathcal{S}_l^{\geq}} \mathcal{T}'_l(\tau)$ 
10:  return  $(\mathcal{T}_l)_{l \geq 0}$ 
11: end procedure

```

Definition 3.4 (Definition 3.1 in [Gø+10]). Ein Wald $\mathcal{T} = \bigcup_{l \geq 0} \mathcal{T}_l$ aus Bäumen mit Wurzel s heißt (α, β) -zuweisbar, wenn gilt:

- Für alle $T \in \mathcal{T}_l$ gilt: $d(T) \leq \alpha 2^l M$
d.h. ein Baum aus \mathcal{T}_l kann mit Geschw. 2^l in $\mathcal{O}(\alpha M)$ besucht werden.

- Für alle $k \geq 1$ gilt: $\sum_{l>k} d(\mathcal{T}_l) \leq \beta M \sum_{l \geq k} 2^l \mu_l$
d.h. die Fahrzeuge mit Geschw. $\geq 2^k$ können den Wald $\mathcal{T}_{>k}$ in $\mathcal{O}(\beta M)$ besuchen.

Lemma 3.5 (Lemma 3.11 in [Gø+10]). *Die von Algorithmus 5 bestimmte Zerlegung $\mathcal{T} = (\mathcal{T}_i)_{i \geq 0}$ ist $(6, 40)$ -zuweisbar.*

3.3 Assignment-Algorithmen

Algorithm 6 FractionalAssignment

```

1: procedure FRACTIONALASSIGNMENT( $(\mathcal{T})$ )
2:    $L := \{T \in \mathcal{T}\}, \quad b(T) := d(T), \quad R := \{i \mid 1 \leq i \leq k\}, \quad b(i) := \beta M 2^{\lambda_i}$ 
3:    $F := \{\{T, i\} \mid T \in \mathcal{T}_l, \lambda_i \geq l - 1\}$ 
4:   Bestimme  $L$ -überdeckendes  $b$ -Matching  $x_{T_i}$ .
5:   return  $(x_{T_i})$ 
6: end procedure

```

Proposition 3.6 (Seite 54 (?) in [Coo+11]). *b -Matching existiert und kann mit Max-Flow in polynomieller Zeit gefunden werden.*

Algorithm 7 RoundingAssignment

```

1: procedure ROUNDINGASSIGNMENT( $(x_{T_i})$ )
2:    $(x'_{T_i}) \leftarrow \text{ROUNDSCHEDULING}(p_{T_i} := \frac{d(T)}{2^{\lambda_i}}, \tilde{x}_{T_i} := \frac{x_{T_i}}{d(T)})$ 
3:    $\tau_i \leftarrow$  Tour durch die Bäume  $T$  mit  $x'_{T_i} = 1$ .
4:   return  $(\tau_i)$ 
5: end procedure

```

Proposition 3.7 (Theorem 1 in [LST90]). *Gegeben folgendes Scheduling-Problem: Job T benötigt auf Maschine i Zeit p_{T_i} . $J_i(t) := \{T \mid p_{T_i} \leq t\}$ und $M_T(t) := \{i \mid p_{T_i} \leq t\}$.*

Eine rationale Lösung \tilde{x}_{T_i} mit

$$\sum_{i \in M_T(t)} \tilde{x}_{T_i} = 1, \quad \sum_{T \in J_i(t)} p_{T_i} \tilde{x}_{T_i} \leq \beta M, \quad \tilde{x}_{T_i} \geq 0$$

kann in polynomieller Zeit zu einer ganzen Lösung x'_{T_i} gerundet werden mit:

$$\sum_{i \in M_T(t)} x'_{T_i} = 1, \quad \sum_{T \in J_i(t)} p_{T_i} x'_{T_i} \leq \beta M + t, \quad x'_{T_i} \in \{0, 1\}$$

Lemma 3.8 (Lemma 3.2 in [Gø+10]). *Gegeben einen (α, β) -zuweisbaren Wald, liefern Algorithmus 6 und Algorithmus 7 eine $(4\alpha + 2\beta)$ -approximative Lösung für **HetTSP**.*

Beweis. \tilde{x}_{Ti} erfüllt:

$$\sum_{i \in M_T(2\alpha M)} \tilde{x}_{Ti} = 1, \quad \sum_{T \in J_i(2\alpha M)} p_{Ti} \tilde{x}_{Ti} \leq \beta M, \quad \tilde{x}_{Ti} \geq 0$$

Somit gibt Algorithmus 7 eine Zuweisung mit

$$\sum_{T \in J_i(t)} p_{Ti} \tilde{x}_{Ti} \leq \beta M + 2\alpha M$$

Durch Verdoppeln der Kanten in den Bäumen, erhalten wir daraus Touren mit $d(\tau_i) \leq 2 \cdot (\beta M + 2\alpha M)$. \square

4 Algorithmus für HetCVRP

Algorithm 8 Reduktion

- 1: **procedure** REDUKTION($G = (V, E), d : E \rightarrow \mathbb{R}_{\geq 0}, s \in V, Q, (\lambda_i), (q_v)$)
 - 2: $V' := \{v^{(j)} \mid v \in V, j = 1, \dots, q_v\}, \quad E' := \{\{v^{(j)}, v\} \mid v \in V, j = 1, \dots, q_v\}$
 - 3: $\tilde{G} := (V \cup V', E \cup E'), \quad \tilde{d}(v^{(j)}, v) := \frac{d(s, v)}{Q}, \text{ sonst wie } d$
 - 4: **return** $(\tilde{G}, \tilde{d}, s, Q, (\mu_i))$
 - 5: **end procedure**
-

Lemma 4.1. Für jede Instanz \mathcal{I} von **HetCVRP** ist $\mathcal{J} := \text{REDUKTION}(\mathcal{I})$ eine Instanz von **HetTSP** mit:

$$\text{OPT}_{\text{HetTSP}}(\mathcal{J}) \in \mathcal{O}(1) \cdot \text{OPT}_{\text{HetCVRP}}(\mathcal{I})$$

Beweis. Sei (σ_i) eine optimale Lösung von \mathcal{I} . Definiere $c_i(v) := \#\text{Einheiten von } i \text{ an } v \text{ geliefert}$. Dann gilt also:

$$\sum_i c_i(v) = q_v, \text{ für alle } v \in V$$

Folglich ist es möglich $U'_i \subseteq V'$ zu wählen, sodass gilt

$$\bigcup_i U'_i = V' \text{ und } \left| \{v^{(j)} \in U'_i\} \right| = c_i(v)$$

Definiere weiter $U_i := \{v \in V \mid v^{(j)} \in U'_i\}$.

Seien nun τ_i minimale **TSP**-Touren durch $U'_i \cup U_i \cup \{s\}$. Zusammen sind diese dann insbesondere eine zulässige Lösung für die **HetTSP**-Instanz \mathcal{J} und es gilt:

$$\begin{aligned} d(\tau_i) &\leq 2 \cdot \text{MST}(U'_i \cup \{s\}) = 2 \cdot \left(\text{MST}(U_i \cup \{s\}) + \sum_{v' \in U'_i} \tilde{d}(v, v') \right) \\ &= 2 \cdot \left(\text{MST}(U_i \cup \{s\}) + \sum_{v \in U_i} c_i(v) \frac{d(s, v)}{Q} \right) \leq 2 \cdot (d(\sigma_i) + d(\sigma_i)) = 4 \cdot d(\sigma_i) \end{aligned}$$

Die erste Abschätzung gilt, da man eine zulässige Tour erhalten kann, indem man einen minimalen Spannbaum nimmt, alle Kanten verdoppelt und dann eine Eulertour bestimmt. Bei der zweiten Abschätzung ist:

- $\text{MST}(U_i \cup \{s\}) \leq d(\sigma_i)$, denn σ_i besucht alle Knoten in U_i , enthält also insbesondere einen Spannbaum von $U_i \cup \{s\}$.
- $\sum_{v \in U_i} c_i(v) \frac{d(s, v)}{Q} \leq d(\sigma_i)$, denn σ_i ist eine zulässige Lösung für das **CVRP**-Problem auf U_i mit Bedarfen $c_i(v)$. Damit folgt die Ungleichung aus Proposition 2.2.

Insgesamt folgt daher:

$$\text{OPT}_{\text{HetTSP}}(\mathcal{J}) \leq \max d(\tau_i) \leq 4 \cdot \max d(\sigma_i) = 4 \cdot \text{OPT}_{\text{HetCVRP}}(\mathcal{I})$$

□

Algorithm 9 Deduktion

```

1: procedure DEDUKTION( $\mathcal{I}, \mathcal{J}, (\tau_i)$ )
2:   for  $i \leftarrow 1, \dots, k$  do
3:      $c_i(v) := \left| \left\{ v^{(j)} \mid v^{(j)} \in \tau_i \right\} \right|$ 
4:      $\sigma_i \leftarrow \text{CVRP-APPROX}(G, d, Q, (c_i(v))_{v \in V})$ 
5:   end for
6:   return  $(\sigma_i)$ 
7: end procedure

```

Lemma 4.2. Für jede Lösung (τ_i) von \mathcal{J} ist (σ_i) eine Lösung von \mathcal{I} mit:

$$\max \frac{d(\sigma_i)}{\lambda_i} \in \mathcal{O}(1) \cdot \max \frac{d(\tau_i)}{\lambda_i}$$

Beweis. Nach Lemma 2.3 liefert Algorithmus 2 eine Lösung mit

$$d(\sigma_i) \leq \rho \cdot \text{OPT}_{\text{TSP}}(U'_i, d) + \frac{2}{Q} \sum_{v' \in U'_i} d'(s, v')$$

Ferner ist

$$\text{OPT}_{\text{TSP}}(U'_i, d) \leq d(\tau_i) \text{ und } \frac{2}{Q} \sum_{v' \in U'_i} d'(s, v') \leq d(\tau_i)$$

Also zusammen

$$d(\sigma_i) \leq (\rho + 1) \cdot d(\tau_i)$$

□

Satz 4.3 (Theorem 4.1 in [Gö+10]). *Es gibt eine $\mathcal{O}(1)$ -approximationserhaltende Reduktion von **HetTSP** auf **HetCVRP**.*

Beweis. Verwende Algorithmus 8 um die Eingabe zu einer Instanz von **HetTSP** zu machen. Finde eine $\mathcal{O}(1)$ -approximative Lösung für diese mit Hilfe von Algorithmus 3. Schließlich erhalte aus dieser mit Algorithmus 9 eine Lösung der **HetCVRP**-Instanz.

Für diese Lösung gilt:

$$\max \frac{d(\sigma_i)}{\lambda_i} \in \mathcal{O}(1) \cdot \max \frac{d(\tau_i)}{\lambda_i} \subseteq \mathcal{O}(1) \cdot \text{OPT}_{\mathbf{HetTSP}}(\mathcal{J}) \subseteq \mathcal{O}(1) \cdot \text{OPT}_{\mathbf{HetCVRP}}(\mathcal{I})$$

□

Literatur

- [Coo+11] W.J. Cook u. a. *Combinatorial Optimization*. Wiley Series in Discrete Mathematics and Optimization. Wiley, 2011. ISBN: 9781118031391. URL: <https://books.google.de/books?id=tarLTNwM3gEC>.
- [FHK76] G. N. Frederickson, M. S. Hecht und C. E. Kim. „Approximation algorithms for some routing problems“. In: *Foundations of Computer Science, 1976., 17th Annual Symposium on*. 1976, S. 216–227. DOI: [10.1109/SFCS.1976.6](https://doi.org/10.1109/SFCS.1976.6).
- [Gø+10] Inge Li Gørtz u. a. „Capacitated Vehicle Routing with Non-Uniform Speeds“. In: *CoRR* abs/1012.1850 (2010). URL: <http://arxiv.org/abs/1012.1850>.
- [HK85] M. Haimovich und A. H. G. Rinnooy Kan. „Bounds and Heuristics for Capacitated Routing Problems“. In: *Mathematics of Operations Research* 10.4 (1985), S. 527–542. ISSN: 0364765X, 15265471. URL: <http://www.jstor.org/stable/3689422>.
- [LST90] Jan Karel Lenstra, David B. Shmoys und Éva Tardos. „Approximation algorithms for scheduling unrelated parallel machines“. In: *Mathematical Programming* 46.1 (1990), S. 259–271. ISSN: 1436-4646. DOI: [10.1007/BF01585745](https://doi.org/10.1007/BF01585745). URL: <http://dx.doi.org/10.1007/BF01585745>.