Effiziente Algorithmen (SS2022) Chapter 6 Approximation I

Walter Unger

Lehrstuhl für Informatik 1

—— 03.06.2022 10:29:31 ——

Contents I

- Einleitung
 - Motivation
 - Cliquenproblem
- Vertex Cover
 - Definition
- Greedy
- TSP und Delta-TSP Einleitung
 - 2-Approximation
- Steiner-Bäume
- 1.5-Approximation

- Einleitung Reduktion
- Approximationsalgorithmus
- Zentrumsproblem
- Einleitung
- Komplexität Approximation
- Färbung Greedy
- Approximation Aussagen

• NP-schwere Probleme doch lösen



- NP-schwere Probleme doch lösen
 - ullet Exakt: " \Longrightarrow " exponentiale Laufzeit



6:1 Motivation 3/7

Einleitung

- NP-schwere Probleme doch lösen
 - Exakt: "⇒" exponentiale Laufzeit
 - Nicht Exakt: "⇒" hoffentlich polynomiale Laufzeit



6:1 Motivation 4/7

Einleitung

NP-schwere Probleme doch lösen.

Exakt: "⇒" exponentiale Laufzeit

Nicht Exakt: "⇒" hoffentlich polynomiale Laufzeit

Hier: Approximationsalgorithmen in polynomialer Laufzeit.



- NP-schwere Probleme doch lösen
 - Exakt: "⇒" exponentiale Laufzeit
 - Nicht Exakt: "⇒" hoffentlich polynomiale Laufzeit
- Hier: Approximationsalgorithmen in polynomialer Laufzeit.
- D.h.: welche Approximationsfaktoren sind möglich?

- NP-schwere Probleme doch lösen
 - Exakt: "⇒" exponentiale Laufzeit
 - Nicht Exakt: "⇒" hoffentlich polynomiale Laufzeit
- Hier: Approximationsalgorithmen in polynomialer Laufzeit.
- D.h.: welche Approximationsfaktoren sind möglich?
- Kommt man beliebig nah an das Optimum?

- NP-schwere Probleme doch lösen
 - Exakt: "⇒" exponentiale Laufzeit
 - Nicht Exakt: "⇒" hoffentlich polynomiale Laufzeit
- Hier: Approximationsalgorithmen in polynomialer Laufzeit.
- D.h.: welche Approximationsfaktoren sind möglich?
- Kommt man beliebig nah an das Optimum?

Definition (Approximation)

 $\mathbb{N}_k = \{\mathbf{1}, \ldots, k\}$

Definition

Sei $L: \mathbb{N} \to \mathbb{R}$ eine Funktion und seien weiter I_n die Eingaben für Algorithmus A. Dann hat A einen Approximationsfaktor, falls gilt:

Definition (Approximation)

$$\mathbb{N}_k = \{\mathbf{1}, \ldots, k\}$$

Definition

Sei $L: \mathbb{N} \to \mathbb{R}$ eine Funktion und seien weiter I_n die Eingaben für Algorithmus A. Dann hat A einen Approximationsfaktor, falls gilt:

• $\forall n \in \mathbb{N} : \forall I \in I_n : \frac{A(I)}{cot(I)} \leqslant L(n)$ bei einem Minimierungsproblem und

Definition (Approximation)

$$\mathbb{N}_k = \{\mathbf{1}, \dots, k\}$$

Definition

Sei $L: \mathbb{N} \to \mathbb{R}$ eine Funktion und seien weiter I_n die Eingaben für Algorithmus A. Dann hat A einen Approximationsfaktor, falls gilt:

- $\forall n \in \mathbb{N} : \forall I \in I_n : \frac{A(I)}{opt(I)} \leqslant L(n)$ bei einem Minimierungsproblem und
- $\forall n \in \mathbb{N} : \forall I \in I_n : \frac{opt(I)}{A(I)} \leqslant L(n)$ bei einem Maximierungsproblem.

Definition

Sei G = (V, E). $C \subset V$ heißt Vertex Cover von G, falls

 $\forall e \in E : C \cap e \neq \emptyset$.

Definition

Sei G = (V, E). $C \subset V$ heißt Vertex Cover von G, falls

 $\forall e \in E : C \cap e \neq \emptyset$.

Definition

Sei G = (V, E). $C \subset V$ heißt Vertex Cover von G, falls

 $\forall e \in E : C \cap e \neq \emptyset$.

• Das folgende Problem ist in \mathcal{NPC} :

Definition 4/10

Einleitung

Definition

Sei G = (V, E). $C \subset V$ heißt Vertex Cover von G, falls

$$\forall e \in E : C \cap e \neq \emptyset.$$

- Das folgende Problem ist in \mathcal{NPC} :
 - Gegeben: G, k.

1 Definition 5/10

Walter Unger 7.10.2024 11:32 SS2022

Einleitung

Definition

Sei G = (V, E). $C \subset V$ heißt Vertex Cover von G, falls

$$\forall e \in E : C \cap e \neq \emptyset.$$

- Das folgende Problem ist in \mathcal{NPC} :
 - Gegeben: G, k.
 - Frage: Gibt es ein Vertex Cover C in G mit $|C| \leq k$.

Definition

Sei G = (V, E). $C \subset V$ heißt Vertex Cover von G, falls

$$\forall e \in E : C \cap e \neq \emptyset.$$

- Das folgende Problem ist in \mathcal{NPC} :
 - Gegeben: G, k.
 - Frage: Gibt es ein Vertex Cover C in G mit $|C| \leq k$.
- Reduktionsidee: $V \setminus C$ ist eine stabile Menge.

Jede Kante muss überdeckt werden.



- Jede Kante muss überdeckt werden.
- Für die Abschätzung des Faktors brauchen wir eine untere Schranke.



- Jede Kante muss überdeckt werden.
- Für die Abschätzung des Faktors brauchen wir eine untere Schranke.
- Für zwei unabhängige Kanten braucht man zwei Knoten zur Überdeckung.



- Jede Kante muss überdeckt werden.
- Für die Abschätzung des Faktors brauchen wir eine untere Schranke.
- Für zwei unabhängige Kanten braucht man zwei Knoten zur Überdeckung.
- Für k paarweise unabhängige Kanten braucht man k Knoten zur Überdeckung.



- Jede Kante muss überdeckt werden.
- Für die Abschätzung des Faktors brauchen wir eine untere Schranke.
- Für zwei unabhängige Kanten braucht man zwei Knoten zur Überdeckung.
- Für k paarweise unabhängige Kanten braucht man k Knoten zur Überdeckung.
- Man weiß aber nicht, welcher Knoten von diesen k Kanten im Cover ist.



- Jede Kante muss überdeckt werden.
- Für die Abschätzung des Faktors brauchen wir eine untere Schranke.
- Für zwei unabhängige Kanten braucht man zwei Knoten zur Überdeckung.
- Für k paarweise unabhängige Kanten braucht man k Knoten zur Überdeckung.
- Man weiß aber nicht, welcher Knoten von diesen k Kanten im Cover ist.
- Idee: Wähle beide für einen Approximationsfaktor von zwei.



- Jede Kante muss überdeckt werden.
- Für die Abschätzung des Faktors brauchen wir eine untere Schranke.
- Für zwei unabhängige Kanten braucht man zwei Knoten zur Überdeckung.
- Für k paarweise unabhängige Kanten braucht man k Knoten zur Überdeckung.
- Man weiß aber nicht, welcher Knoten von diesen k Kanten im Cover ist.
- Idee: Wähle beide für einen Approximationsfaktor von zwei.



Theorem



Theorem

Das Vertex Cover Problem kann mit einem Faktor von 2 approximiert werden.

9 Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G = (V, E).

Theorem

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G = (V, E).
- **2** Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.

Theorem

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G = (V, E).
- ② Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.

Theorem

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G = (V, E).
- **2** Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
- Zeige Korrektheit:

Theorem

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G = (V, E).
- **2** Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
- Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.

Theorem

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G = (V, E).
- **2** Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
 - Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.
 - Damit ist *M* kein inklusions-maximales Matching, Widerspruch.

$\mathsf{Theorem}$

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G=(V,E).
- ② Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
 - Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.
 - Damit ist M kein inklusions-maximales Matching, Widerspruch.
 - Zeige Approximationsfaktor: Setze $\tau(G) = |V| \alpha(G)$ und

$\mathsf{Theorem}$

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G=(V,E).
- ② Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
- Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.
 - Damit ist M kein inklusions-maximales Matching, Widerspruch.
- Zeige Approximationsfaktor: Setze $\tau(G) = |V| \alpha(G)$ und
 - M_{max}(G) Maximales Matching von G.

Einleitung

$\mathsf{Theorem}$

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G=(V,E).
- ② Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
- Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.
 - Damit ist M kein inklusions-maximales Matching, Widerspruch.
- Zeige Approximationsfaktor: Setze $\tau(G) = |V| \alpha(G)$ und
 - M_{max}(G) Maximales Matching von G.
 - Schätze Approximationsfaktor ab:

$$\frac{2 \cdot |M|}{\tau(G)} \leqslant \frac{2 \cdot |M_{\sf max}(G)|}{\tau(G)}$$
 beachte: $|M_{\sf max}(G)| \geqslant |M|$

Einleitung

$\mathsf{Theorem}$

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G=(V,E).
- ② Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
 - Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.
 - Damit ist M kein inklusions-maximales Matching, Widerspruch.
 - Zeige Approximationsfaktor: Setze $\tau(G) = |V| \alpha(G)$ und
 - M_{max}(G) Maximales Matching von G.
 - Schätze Approximationsfaktor ab:

$$\frac{2 \cdot |M|}{\tau(G)} \leqslant \frac{2 \cdot |M_{\text{max}}(G)|}{\tau(G)}$$
 beachte: $|M_{\text{max}}(G)| \geqslant |M|$

Einleitung

Vertex Cover

$\mathsf{Theorem}$

Das Vertex Cover Problem kann mit einem Faktor von 2 approximiert werden.

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G=(V,E).
- ② Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
 - Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.
 - Damit ist M kein inklusions-maximales Matching, Widerspruch.
 - Zeige Approximationsfaktor: Setze $\tau(G) = |V| \alpha(G)$ und
 - M_{max}(G) Maximales Matching von G.

 - Schätze Approximationsfaktor ab:

$$\begin{array}{lll} \frac{2\cdot|M|}{\tau(G)} & \leqslant & \frac{2\cdot|M_{\max}(G)|}{\tau(G)} & \text{beachte: } |M_{\max}(G)| \geqslant |M| \\ & \leqslant & \frac{2\cdot|M_{\max}(G)|}{|M_{\max}(G)|} & \text{beachte: } |M_{\max}(G)| \leqslant \tau(G) \end{array}$$

Zentrumsproblem

6:16 Greedy 13/13 Vertex Cover

Einleitung

$\mathsf{Theorem}$

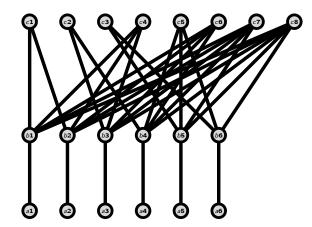
Das Vertex Cover Problem kann mit einem Faktor von 2 approximiert werden.

- **9** Bestimme inklusions-maximales Matching $M \subset E$ auf Eingabegraph G=(V,E).
- ② Wähle Vertex Cover $C = \bigcup_{e \in M} e$ und gib C aus.
- Zeige Korrektheit:
 - Angenommen $C = \bigcup_{e \in M} e$ ist kein Vertex Cover, d.h. $\exists e : e \cap C = \emptyset$.
 - Damit ist M kein inklusions-maximales Matching, Widerspruch.
- Zeige Approximationsfaktor: Setze $\tau(G) = |V| \alpha(G)$ und
 - M_{max}(G) Maximales Matching von G.
 - Schätze Approximationsfaktor ab:

$$\begin{array}{lll} \frac{2\cdot|M|}{\tau(G)} & \leqslant & \frac{2\cdot|M_{\max}(G)|}{\tau(G)} & \text{beachte: } |M_{\max}(G)| \geqslant |M| \\ & \leqslant & \frac{2\cdot|M_{\max}(G)|}{|M_{\max}(G)|} & \text{beachte: } |M_{\max}(G)| \leqslant \tau(G) \\ & = & 2 \end{array}$$

Walter Unger 7.10.2024 11:32 SS2022

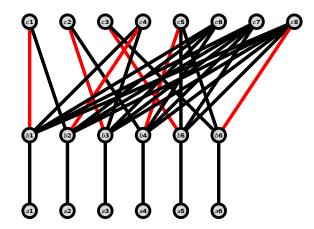
Beispiel





Walter Unger 7.10.2024 11:32 SS2022

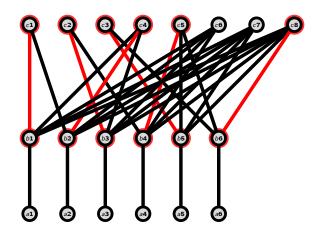
Beispiel





Walter Unger 7.10.2024 11:32 SS2022

Beispiel





6:18 Greedy 1/7

Vertex Cover

Theorem (Monien)

Das Vertex Cover Problem kann mit einem Faktor von 2 $-\frac{\log\log n}{2\cdot\log n}$ approximiert werden.

Vertex Cover

Theorem (Monien)

Das Vertex Cover Problem kann mit einem Faktor von 2 $-\frac{\log\log n}{2\cdot\log n}$ approximiert werden.

Zusammenfassung:



Vertex Cover

Theorem (Monien)

Das Vertex Cover Problem kann mit einem Faktor von 2 $-\frac{\log \log n}{2 \cdot \log n}$ werden.

- Zusammenfassung:
- Vertex Cover ist mit Faktor 2 approximierbar.



Theorem (Monien)

Das Vertex Cover Problem kann mit einem Faktor von 2 $-\frac{\log\log n}{2\cdot\log n}$ approximiert werden.

- Zusammenfassung:
- Vertex Cover ist mit Faktor 2 approximierbar.
- Clique ist mit keinem konstanten Faktor approximierbar.

Vantau Causan

Vertex Cover

Theorem (Monien)

Das Vertex Cover Problem kann mit einem Faktor von $2-\frac{\log\log n}{2\cdot\log n}$ approximiert werden.

- Zusammenfassung:
- Vertex Cover ist mit Faktor 2 approximierbar.
- Clique ist mit keinem konstanten Faktor approximierbar.
- Independent Set ist mit keinem konstanten Faktor approximierbar.

Theorem (Monien)

Das Vertex Cover Problem kann mit einem Faktor von 2 $-\frac{\log\log n}{2\cdot\log n}$ approximiert werden.

- Zusammenfassung:
- Vertex Cover ist mit Faktor 2 approximierbar.
- Clique ist mit keinem konstanten Faktor approximierbar.
- Independent Set ist mit keinem konstanten Faktor approximierbar.
- NP-vollständige Probleme unterscheiden sich in ihrer Approximierbarkeit.

Vertex Cover

Theorem (Monien)

Das Vertex Cover Problem kann mit einem Faktor von $2 - \frac{\log \log n}{2 \cdot \log n}$ approximiert werden.

- Zusammenfassung:
- Vertex Cover ist mit Faktor 2 approximierbar.
- Clique ist mit keinem konstanten Faktor approximierbar.
- Independent Set ist mit keinem konstanten Faktor approximierbar.
- \mathcal{NP} -vollständige Probleme unterscheiden sich in ihrer Approximierbarkeit.

6:19 Einleitung 1/10

Walter Unger 7.10.2024 11:32 SS2022

TSP

Definition (TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.



6:19 Einleitung 2/10

TSP

Definition (TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

Einleitung

TSP

Definition (TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$. D.h. ein Kreis, der jeden Knoten genau einmal besucht.

• Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).

Einleitung 4/

Walter Unger 7.10.2024 11:32 SS2022

TSP

Definition (TSP)

Gegeben $G=(V,E),\ L\in\mathbb{Q}$ und $c:E\mapsto\mathbb{Q}.$ Bestimme spannenden einfachen Kreis C mit $c(C)\leqslant L.$ D.h. ein Kreis, der jeden Knoten genau einmal besucht.

- \bullet Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
 - G = (V, E) Eingabe für Hamilton Kreis.

Einleitung 5/10

Walter Unger 7.10.2024 11:32 SS2022

TSP

Definition (TSP)

Gegeben G=(V,E), $L\in\mathbb{Q}$ und $c:E\mapsto\mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C)\leqslant L$. D.h. ein Kreis, der jeden Knoten genau einmal besucht.

- ullet Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:

Definition (TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$. D.h. ein Kreis, der jeden Knoten genau einmal besucht.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und

Definition (TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$. D.h. ein Kreis, der jeden Knoten genau einmal besucht.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - c(e) = 2 falls $e \notin E$.

Definition (TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - c(e) = 2 falls $e \notin E$.
 - Falls G einen Hamiltonkreis hat, dann G' Tour mit Kosten n.

6:19 Einleitung 9/10

Walter Unger 7.10.2024 11:32 SS2022

TSP

Definition (TSP)

Gegeben G = (V, E), $L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$. Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - c(e) = 2 falls $e \notin E$.
 - Falls G einen Hamiltonkreis hat, dann G' Tour mit Kosten n.
 - Anderenfalls hat G' eine Tour mit Kosten $\geq n+1$.

6:19 Einleitung 10/10

Walter Unger 7.10.2024 11:32 SS2022

TSP

Definition (TSP)

Gegeben G = (V, E), $L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leqslant L$.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - c(e) = 2 falls $e \notin E$.
 - Falls G einen Hamiltonkreis hat, dann G' Tour mit Kosten n.
 - Anderenfalls hat G' eine Tour mit Kosten $\geq n+1$.

6:20 Einleitung 1/7

Walter Unger 7.10.2024 11:32

TSP

Theorem

6:20 Einleitung 2/7 Walter Unger 7.10.2024 11:32

TSP

Theorem

Sei $\alpha(n)$ eine polynomialzeit-berechenbare Funktion, G = (V, E) und n = |V|. Falls $\mathcal{P} \neq \mathcal{NP}$, gilt: TSP kann nicht in Polynomialzeit mit Faktor $\alpha(n)$ approximiert werden.

• Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
- Gleiche Reduktion zeigt obigen Satz:
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
- Gleiche Reduktion zeigt obigen Satz:
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - $c(e) = \alpha(n) \cdot n + 1$ falls $e \notin E$.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
- Gleiche Reduktion zeigt obigen Satz:
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - $c(e) = \alpha(n) \cdot n + 1$ falls $e \notin E$.
 - Falls G einen Hamiltonkreis hat, dann G' Tour mit Kosten n.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
- Gleiche Reduktion zeigt obigen Satz:
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - $c(e) = \alpha(n) \cdot n + 1$ falls $e \notin E$.
 - Falls G einen Hamiltonkreis hat, dann G' Tour mit Kosten n.
 - Anderenfalls hat G' eine Tour mit Kosten $\geq n-1+\alpha(n)\cdot n$.

- Problem ist in \mathcal{NPC} (Reduktion von Hamilton Kreis).
- Gleiche Reduktion zeigt obigen Satz:
 - G = (V, E) Eingabe für Hamilton Kreis.
 - Eingabe für TSP ist Clique C = (V, E') mit:
 - c(e) = 1 falls $e \in E$ und
 - $c(e) = \alpha(n) \cdot n + 1$ falls $e \notin E$.
 - Falls G einen Hamiltonkreis hat, dann G' Tour mit Kosten n.
 - Anderenfalls hat G' eine Tour mit Kosten $\geq n-1+\alpha(n)\cdot n$.

6:21 Einleitung 1/7

 Δ -TSP

Definition (Δ -TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$

• mit: $\forall v, w \in V : e = \{v, w\} \in E, c(e) \geq 0$ und

• mit: $\forall v, w, z \in V : c(v, z) \leq c(v, w) + c(w, z)$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

Δ-TSP

Definition (Δ-TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$

• mit: $\forall v, w \in V : e = \{v, w\} \in E, c(e) \geqslant 0$ und

• mit: $\forall v, w, z \in V : c(v, z) \leqslant c(v, w) + c(w, z)$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

Definition (Δ -TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$

• mit: $\forall v, w \in V : e = \{v, w\} \in E, c(e) \geq 0$ und

• mit: $\forall v, w, z \in V : c(v, z) \leq c(v, w) + c(w, z)$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

• Obige Eigenschaft auf c wird Metrik genannt.

Definition (Δ -TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$

• mit: $\forall v, w \in V : e = \{v, w\} \in E, c(e) \geq 0$ und

• mit: $\forall v, w, z \in V : c(v, z) \leq c(v, w) + c(w, z)$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

• Obige Eigenschaft auf c wird Metrik genannt.

Motivation: Die Abstände von Punkten in der Ebene bilden eine Metrik.



Definition (Δ -TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$

• mit: $\forall v, w \in V : e = \{v, w\} \in E, c(e) \geq 0$ und

• mit: $\forall v, w, z \in V : c(v, z) \leq c(v, w) + c(w, z)$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

- Obige Eigenschaft auf c wird Metrik genannt.
- Motivation: Die Abstände von Punkten in der Ebene bilden eine Metrik.

Definition (Δ -TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$

- mit: $\forall v, w \in V : e = \{v, w\} \in E, c(e) \geq 0$ und
- mit: $\forall v, w, z \in V : c(v, z) \leq c(v, w) + c(w, z)$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

- Obige Eigenschaft auf c wird Metrik genannt.
- Motivation: Die Abstände von Punkten in der Ebene bilden eine Metrik.

Theorem

 Δ -TSP mit Gewichtsfunktion $c \mapsto \{1,2\}$ ist in \mathcal{NPC} .

Beweis: Reduktion von Hamilton Kreis auf planare Graphen.

Definition (Δ -TSP)

Gegeben $G = (V, E), L \in \mathbb{Q}$ und $c : E \mapsto \mathbb{Q}$

- mit: $\forall v, w \in V : e = \{v, w\} \in E, c(e) \geq 0$ und
- mit: $\forall v, w, z \in V : c(v, z) \leq c(v, w) + c(w, z)$.

Bestimme spannenden einfachen Kreis C mit $c(C) \leq L$.

- Obige Eigenschaft auf c wird Metrik genannt.
- Motivation: Die Abstände von Punkten in der Ebene bilden eine Metrik.

Theorem

 Δ -TSP mit Gewichtsfunktion $c \mapsto \{1,2\}$ ist in \mathcal{NPC} .

Beweis: Reduktion von Hamilton Kreis auf planare Graphen.

Suche untere Schranke:



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke.



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke.
 - Idee: Wähle minimalen Spannbaum!



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke.
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke.
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.
 - Damit kann ein beliebiger Kreis, der jeden Knoten mindestens einmal besucht, zu einer kostengünstigeren TSP-Tour umgeformt werden.



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.
 - Damit kann ein beliebiger Kreis, der jeden Knoten mindestens einmal besucht, zu einer kostengünstigeren TSP-Tour umgeformt werden.
 - Idee: Konstruiere daher einen Eulerkreis (besucht alle Kanten).



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.
 - Damit kann ein beliebiger Kreis, der jeden Knoten mindestens einmal besucht, zu einer kostengünstigeren TSP-Tour umgeformt werden.
 - Idee: Konstruiere daher einen Eulerkreis (besucht alle Kanten).
- G = (V, E) ist Eulergraph (enthält Eulerkreis):



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke.
 - Idee: Wähle minimalen Spannbaum!
 - Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.
 - Damit kann ein beliebiger Kreis, der jeden Knoten mindestens einmal besucht, zu einer kostengünstigeren TSP-Tour umgeformt werden.
 - Idee: Konstruiere daher einen Eulerkreis (besucht alle Kanten).
 - G = (V, E) ist Eulergraph (enthält Eulerkreis):
 - G ist zusammenhängend.



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.
 - Damit kann ein beliebiger Kreis, der jeden Knoten mindestens einmal besucht, zu einer kostengünstigeren TSP-Tour umgeformt werden.
 - Idee: Konstruiere daher einen Eulerkreis (besucht alle Kanten).
- G = (V, E) ist Eulergraph (enthält Eulerkreis):
 - G ist zusammenhängend.
 - Alle Knoten haben geraden Grad.



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke.
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.
 - Damit kann ein beliebiger Kreis, der jeden Knoten mindestens einmal besucht, zu einer kostengünstigeren TSP-Tour umgeformt werden.
 - Idee: Konstruiere daher einen Eulerkreis (besucht alle Kanten).
- G = (V, E) ist Eulergraph (enthält Eulerkreis):
 - G ist zusammenhängend.
 - Alle Knoten haben geraden Grad.
- Kombination: Mit Hilfe des Spannbaums wird Eulerkreis konstruiert.



- Suche untere Schranke:
 - TSP-Tour ist ein spannender Graph (enthält alle Knoten)
 - Jeder kostenminimale zusammenhängende Graph bildet eine untere Schranke.
 - Idee: Wähle minimalen Spannbaum!
- Nutze Dreiecksungleichung aus:
 - D.h. jede Abkürzung verringert die Kosten.
 - Damit kann ein beliebiger Kreis, der jeden Knoten mindestens einmal besucht, zu einer kostengünstigeren TSP-Tour umgeformt werden.
 - Idee: Konstruiere daher einen Eulerkreis (besucht alle Kanten).
- G = (V, E) ist Eulergraph (enthält Eulerkreis):
 - G ist zusammenhängend.
 - Alle Knoten haben geraden Grad.
- Kombination: Mit Hilfe des Spannbaums wird Eulerkreis konstruiert.



6:23 2-Approximation 1/8

Approximation

Theorem



6:23 2-Approximation 2/8

Walter Unger 7.10.2024 11:32 SS2022

Approximation

Theorem

 Δ -TSP ist mit einem Faktor von 2 in Zeit $O(n^2 \log n)$ approximierbar.

Bestimme minimalen Spannbaum T von G.



Approximation

Theorem

- Bestimme minimalen Spannbaum T von G.
- 4 Verdoppele die Kanten von T und erzeuge Graphen T'.

Theorem

- Bestimme minimalen Spannbaum T von G.
- \bigcirc Verdoppele die Kanten von T und erzeuge Graphen T'.
- \odot Damit sind alle Knotengrade in T' gerade (da verdoppelt).

Theorem

- Bestimme minimalen Spannbaum T von G.
- ② Verdoppele die Kanten von T und erzeuge Graphen T'.
- ② Damit sind alle Knotengrade in T' gerade (da verdoppelt).
- \bigcirc Bestimme in T' einen Euler-Kreis C'.

$\mathsf{Theorem}$

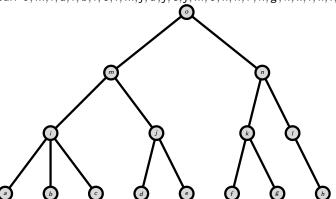
- Bestimme minimalen Spannbaum T von G.
- \bigcirc Verdoppele die Kanten von T und erzeuge Graphen T'.
- \odot Damit sind alle Knotengrade in T' gerade (da verdoppelt).
- \bigcirc Bestimme in T' einen Fuler-Kreis C'.
- Verkürze C' durch Überspringen doppelter Knoten.

$\mathsf{Theorem}$

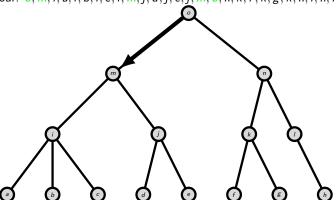
- Bestimme minimalen Spannbaum T von G.
- \bigcirc Verdoppele die Kanten von T und erzeuge Graphen T'.
- \odot Damit sind alle Knotengrade in T' gerade (da verdoppelt).
- \bigcirc Bestimme in T' einen Fuler-Kreis C'.
- Verkürze C' durch Überspringen doppelter Knoten.
- Oadurch entsteht Kreis C, gebe C aus.

$\mathsf{Theorem}$

- Bestimme minimalen Spannbaum T von G.
- \bigcirc Verdoppele die Kanten von T und erzeuge Graphen T'.
- \odot Damit sind alle Knotengrade in T' gerade (da verdoppelt).
- \bigcirc Bestimme in T' einen Fuler-Kreis C'.
- Verkürze C' durch Überspringen doppelter Knoten.
- Oadurch entsteht Kreis C, gebe C aus.

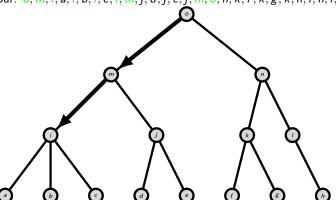


Beispiel



2-Approximation 3/16 Walter Unger 7.10.2024 11:32 \$\$2022

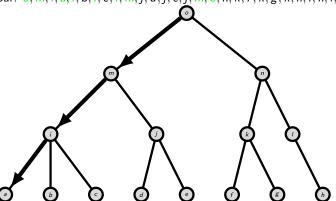
Beispiel



2-Approximation 4/16

Walter Unger 7.10.2024 11:32

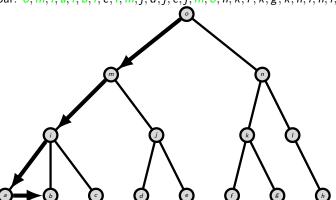
Beispiel



2-Approximation 5/16

Walter Unger 7.10.2024 11:32 SS2022

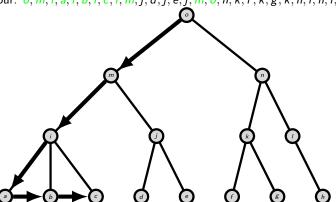
Beispiel



2-Approximation 6/16

Walter Unger 7.10.2024 11:32

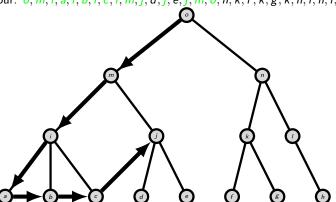
Beispiel



Walter Unger 7.10.2024 11:32

2-Approximation 7/16

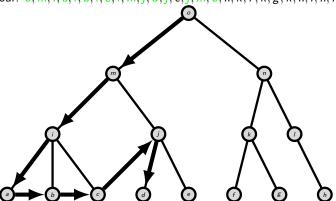
Beispiel



2-Approximation 8/16

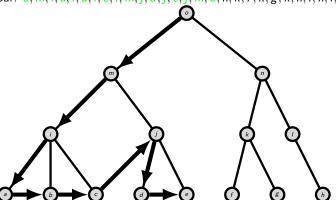
Walter Unger 7.10.2024 11:32

Beispiel

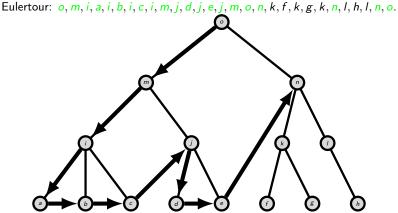


Walter Unger 7.10.2024 11:32

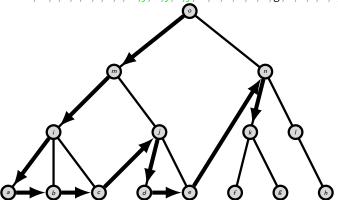
Beispiel

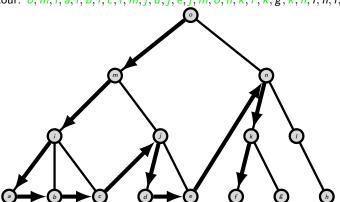


Bestimme Eulertour: Gehe einmal inorder um den Baum herum.



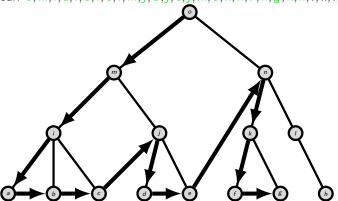
Beispiel





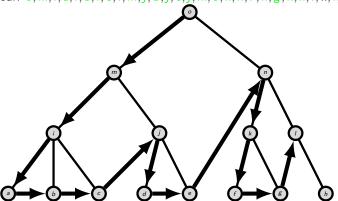
Walter Unger 7.10.2024 11:32

Beispiel





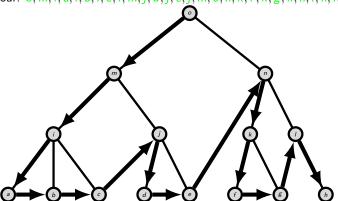
Beispiel



15/16

Walter Unger 7.10.2024 11:32

Beispiel

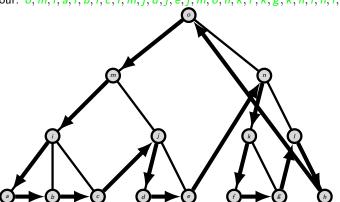


16/16

Walter Unger 7.10.2024 11:32

Beispiel

Bestimme Eulertour: Gehe einmal inorder um den Baum herum. Eulertour: o, m, i, a, i, b, i, c, i, m, j, d, j, e, j, m, o, n, k, f, k, g, k, n, l, h, l, n, o.



Approximation (Beweis)

G - e ist G ohne eine beliebige Kante

Seien T minimaler Spannbaum von G,
 T' mit verdoppelte Kanten,
 C' Eulerkreis in T' und
 C gefundene Lösung.



G - e ist G ohne eine beliebige Kante

Approximation (Beweis)

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Eulerkreis in T' und C gefundene Lösung.
 - Sei C* eine minimale TSP Tour.

G - e ist G ohne eine beliebige Kante

Färbung

Einleitung

000000000

Approximation (Beweis)

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
 - Sei C* eine minimale TSP Tour.
 - $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.

Approximation (Beweis)

Einleitung

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.

Approximation (Beweis)

Einleitung

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
 - Sei C* eine minimale TSP Tour.
 - $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.
 - $c(C) \leqslant c(C') = 2 \cdot c(T)$.

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
 - Sei C* eine minimale TSP Tour.
 - $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.
 - $c(C) \leqslant c(C') = 2 \cdot c(T)$.
 - Da die Kanten verdoppelt worden sind.

Approximation (Beweis)

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.
- $c(C) \leqslant c(C') = 2 \cdot c(T)$.
 - Da die Kanten verdoppelt worden sind.
- $\frac{c(C)}{c(C^*)}$

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
 - Sei C* eine minimale TSP Tour.
 - $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.
 - $c(C) \leqslant c(C') = 2 \cdot c(T)$.
 - Da die Kanten verdoppelt worden sind.
- $\frac{c(C)}{c(C^*)} \leqslant \frac{2 \cdot c(T)}{c(C^*)}$

Approximation (Beweis)

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
 - Sei C* eine minimale TSP Tour.
 - $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.
 - $c(C) \leqslant c(C') = 2 \cdot c(T)$.
 - Da die Kanten verdoppelt worden sind.
- $\frac{c(C)}{c(C^*)} \leqslant \frac{2 \cdot c(T)}{c(C^*)} \leqslant \frac{2 \cdot c(C^*)}{c(C^*)} = 2$

G - e ist G ohne eine beliebige Kante

Approximation (Beweis)

Einleitung

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leq c(C^* e) \leq c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.
- $c(C) \leqslant c(C') = 2 \cdot c(T)$.
 - Da die Kanten verdoppelt worden sind.
- $\frac{c(C)}{c(C^*)} \leqslant \frac{2 \cdot c(T)}{c(C^*)} \leqslant \frac{2 \cdot c(C^*)}{c(C^*)} = 2$
- Laufzeit O(n² log n).

G - e ist G ohne eine beliebige Kante

Färbung

Einleitung

Approximation (Beweis)

- Seien T minimaler Spannbaum von G, T' mit verdoppelte Kanten, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leq c(C^* e) \leq c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum.
- $c(C) \leqslant c(C') = 2 \cdot c(T)$.
 - Da die Kanten verdoppelt worden sind.
- $\frac{c(C)}{c(C^*)} \leqslant \frac{2 \cdot c(T)}{c(C^*)} \leqslant \frac{2 \cdot c(C^*)}{c(C^*)} = 2$
- Laufzeit O(n² log n).

 Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.



- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls *v* bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.



- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls v bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.

- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls v bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.

- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls *v* bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.



 Einleitung
 Vertex Cover
 TSP und Delta-TSP
 Steiner-Bäume
 Zentrumsproblem
 Färbung

 000000000
 00000000
 000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 000000000000
 000000000000
 000000000000
 000000000000
 000000000000
 000000000000
 0000000000000
 000000000000
 00000000000000
 000000000000
 000000000000
 000000000000
 000000000000
 000000000000
 00000000000
 000000000000
 000000000000
 000000000000
 000000000000
 000000000000
 00000000000
 00000000000
 0000000000
 00000000000
 0000000000
 0000000000
 0000000000
 0000000000
 000000000
 0000000000
 000000000000
 000000000
 0000000000
 0000000000
 000000000
 000000000
 000000000
 00000000000
 000000000
 0000000000
 000000000
 0000000000
 0000000000
 00000000000
 0000000000
 0000000000
 00000000000

- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls *v* bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.



- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls v bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.
- Geht nur, wenn die Anzahl der ungeraden Knoten gerade ist:

- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls *v* bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.
- Geht nur, wenn die Anzahl der ungeraden Knoten gerade ist:
 - Zählen wir die Kombinationen X: Knoten v inzident zu Kante {v, w}:



- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls *v* bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.
- Geht nur, wenn die Anzahl der ungeraden Knoten gerade ist:
 - Zählen wir die Kombinationen X:
 Knoten v inzident zu Kante {v, w}:
 - Aus der Sicht der Kanten: $|X| = 2 \cdot |E|$.



- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls v bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.
- Geht nur, wenn die Anzahl der ungeraden Knoten gerade ist:
 - 7ählen wir die Kombinationen X. Knoten v inzident zu Kante $\{v, w\}$:
 - Aus der Sicht der Kanten: |X| = 2 · |E|.
 - Aus der Sicht der Knoten: $|X| = \sum_{v \in V} \delta(v)$.



- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls v bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.
- Geht nur, wenn die Anzahl der ungeraden Knoten gerade ist:
 - 7ählen wir die Kombinationen X. Knoten v inzident zu Kante $\{v, w\}$:
 - Aus der Sicht der Kanten: |X| = 2 · |E|.
 - Aus der Sicht der Knoten: $|X| = \sum_{v \in V} \delta(v)$.
 - Sei U die Menge der Knoten mit ungeradem Grad.



Einleitung

- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls v bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.
- Geht nur, wenn die Anzahl der ungeraden Knoten gerade ist:
 - 7ählen wir die Kombinationen X. Knoten v inzident zu Kante $\{v, w\}$:
 - Aus der Sicht der Kanten: $|X| = 2 \cdot |E|$.
 - Aus der Sicht der Knoten: $|X| = \sum_{v \in V} \delta(v)$.
 - Sei *U* die Menge der Knoten mit ungeradem Grad.
 - $2 \cdot |E| = \sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v)$.

- Durch die Verdopplung aller Kanten des Spannbaums bekommen wir den Faktor von 2.
- D.h.: Falls v bereits gerade ist, wird trotzdem der Knotengrad verdoppelt.
- Daher: "verdoppele" den Knotengrad nur von den ungeraden Knoten.
- Oder noch besser: Erhöhe den Knotengrad der ungeraden Knoten um eins.
- D.h. jeder ungerade Knoten bekommt eine zusätzliche Kante.
- Das entspricht einem Matching zwischen den Knoten vom ungeraden Grad.
- Geht nur, wenn die Anzahl der ungeraden Knoten gerade ist:
 - 7ählen wir die Kombinationen X. Knoten v inzident zu Kante $\{v, w\}$:
 - Aus der Sicht der Kanten: $|X| = 2 \cdot |E|$.
 - Aus der Sicht der Knoten: $|X| = \sum_{v \in V} \delta(v)$.
 - Sei U die Menge der Knoten mit ungeradem Grad.
 - $2 \cdot |E| = \sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v)$.



1.5-Approximation

Theorem



6:27 1.5-Approximation 2/10

1.5-Approximation

Theorem

 Δ -TSP ist mit einem Faktor von 1.5 in Zeit $O(n^3)$ approximierbar.

• Bestimme minimalen Spannbaum T von G.



6:27 1.5-Approximation 3/10

1.5-Approximation

Theorem

- Bestimme minimalen Spannbaum T von G.
- Seien U die Knoten von ungeraden Grad in T.



1.5-Approximation

Einleitung

000000000

Theorem

- Bestimme minimalen Spannbaum T von G.
- Seien U die Knoten von ungeraden Grad in T.
- Beachte |U| ist gerade: $\sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v) = 2|E|$.

Theorem

- Bestimme minimalen Spannbaum T von G.
- ullet Seien U die Knoten von ungeraden Grad in T.
- Beachte |U| ist gerade: $\sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v) = 2|E|$.
- ullet Bestimme kostenminimales Matching M zwischen den Knoten aus U.

1.5-Approximation

Einleitung

Theorem

- Bestimme minimalen Spannbaum T von G.
- Seien U die Knoten von ungeraden Grad in T.
- Beachte |U| ist gerade: $\sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v) = 2|E|$.
- ullet Bestimme kostenminimales Matching M zwischen den Knoten aus U.
- Bestimme in $T \cup M$ einen Eulerkreis C'.

Theorem

- Bestimme minimalen Spannbaum T von G.
- Seien U die Knoten von ungeraden Grad in T.
- Beachte |U| ist gerade: $\sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v) = 2|E|$.
- ullet Bestimme kostenminimales Matching M zwischen den Knoten aus U.
- Bestimme in $T \cup M$ einen Eulerkreis C'.
- Verkürze C' durch Überspringen doppelter Knoten.

$\mathsf{Theorem}$

- Bestimme minimalen Spannbaum T von G.
- Seien U die Knoten von ungeraden Grad in T.
- Beachte |U| ist gerade: $\sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v) = 2|E|$.
- Bestimme kostenminimales Matching M zwischen den Knoten aus U.
- Bestimme in $T \cup M$ einen Eulerkreis C'.
- Verkürze C' durch Überspringen doppelter Knoten.
- Dadurch entsteht Kreis C.

1.5-Approximation

Einleitung

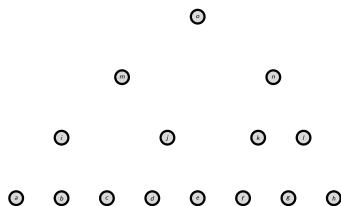
$\mathsf{Theorem}$

- Bestimme minimalen Spannbaum T von G.
- Seien U die Knoten von ungeraden Grad in T.
- Beachte |U| ist gerade: $\sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v) = 2|E|$.
- Bestimme kostenminimales Matching M zwischen den Knoten aus U.
- Bestimme in $T \cup M$ einen Eulerkreis C'.
- Verkürze C' durch Überspringen doppelter Knoten.
- Dadurch entsteht Kreis C.
- Gebe C aus.

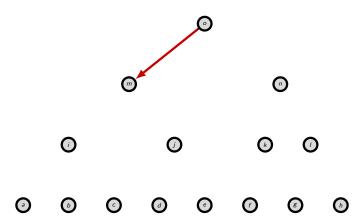
Theorem

- Bestimme minimalen Spannbaum T von G.
- Seien U die Knoten von ungeraden Grad in T.
- Beachte |U| ist gerade: $\sum_{v \in U} \delta(v) + \sum_{v \notin U} \delta(v) = 2|E|$.
- Bestimme kostenminimales Matching M zwischen den Knoten aus U.
- Bestimme in $T \cup M$ einen Eulerkreis C'.
- Verkürze C' durch Überspringen doppelter Knoten.
- Dadurch entsteht Kreis C.
- Gebe C aus.

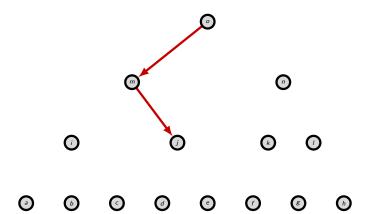
Beispiel (Bestimmen der Eulertour)



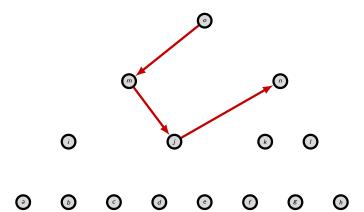




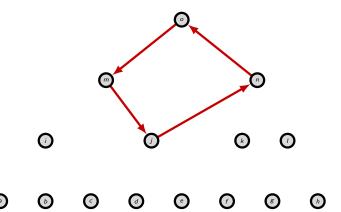




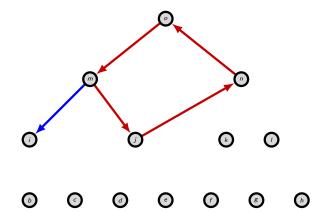








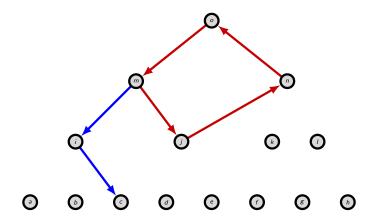




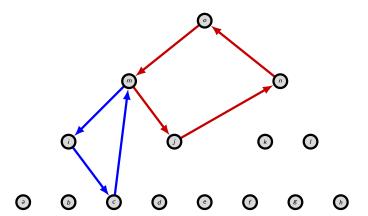


6:28 1.5-Approximation 7/41

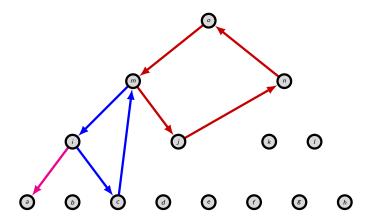
Walter Unger 7.10.2024 11:32 SS2022



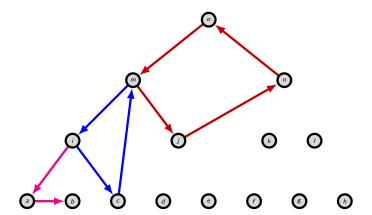






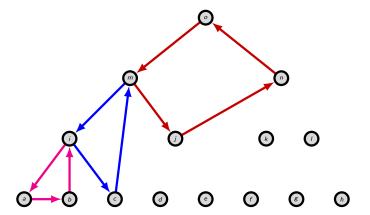








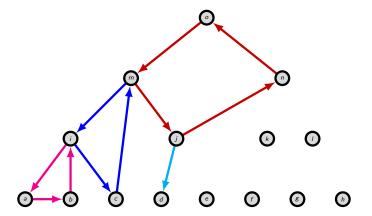
6:28 1.5-Approximation 11/41 Walter Unger 7.10.2024 11:32 SS2022





6:28 1.5-Approximation 12/41

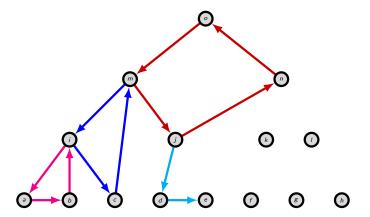
Walter Unger 7.10.2024 11:32





6:28 1.5-Approximation 13/41

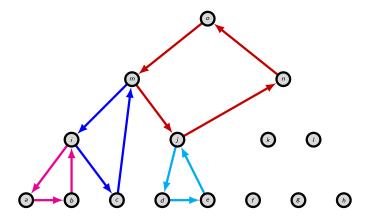
Walter Unger 7.10.2024 11:32





6:28 1.5-Approximation 14/41

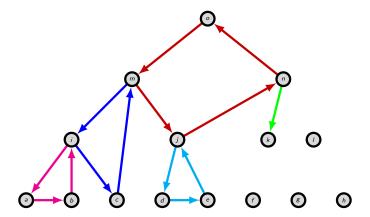
Walter Unger 7.10.2024 11:32





6:28 1.5-Approximation 15/41

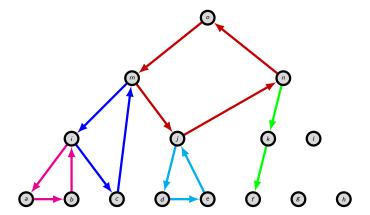
Walter Unger 7.10.2024 11:32



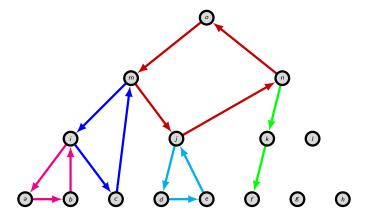


6:28 1.5-Approximation 16/41

Walter Unger 7.10.2024 11:32



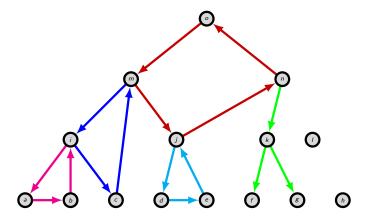




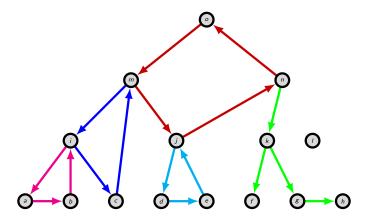


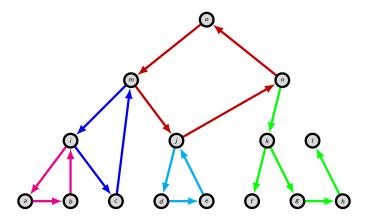
6:28 1.5-Approximation 18/41

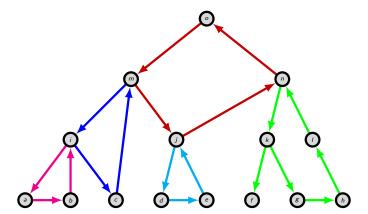
Walter Unger 7.10.2024 11:32



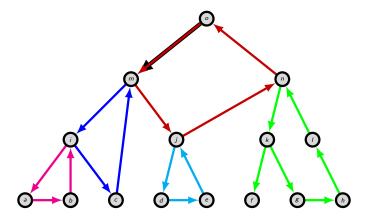


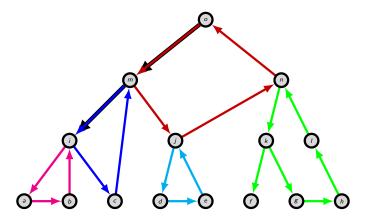






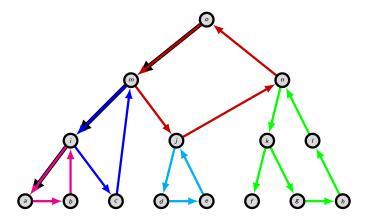






6:28 1.5-Approximation 24/41

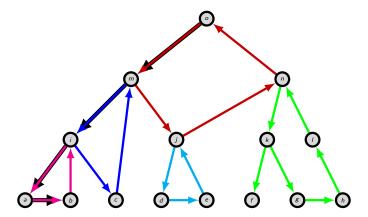
Walter Unger 7.10.2024 11:32





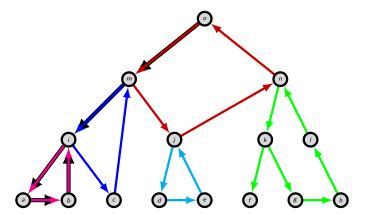
6:28 1.5-Approximation 25/41

Walter Unger 7.10.2024 11:32



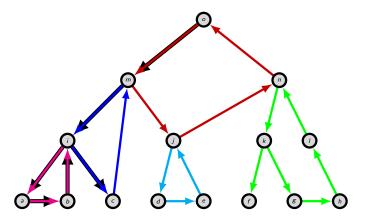
6:28 1.5-Approximation 26/41

Walter Unger 7.10.2024 11:32



6:28 1.5-Approximation 27/41

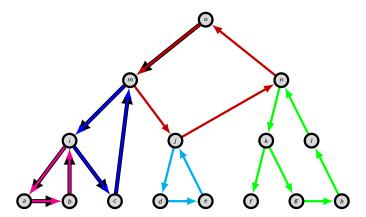
Walter Unger 7.10.2024 11:32





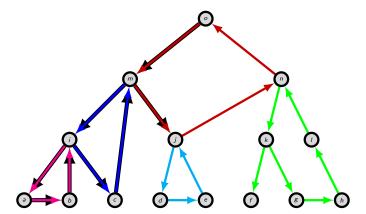
6:28 1.5-Approximation 28/41

Walter Unger 7.10.2024 11:32



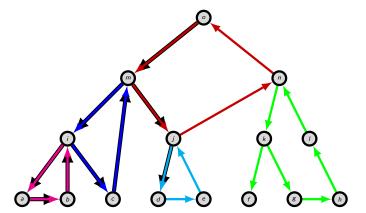


6:28 1.5-Approximation 29/41 Walter Unger 7.10.2024 11:32



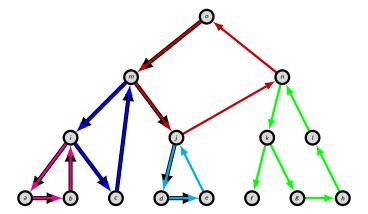


6:28 1.5-Approximation 30/41



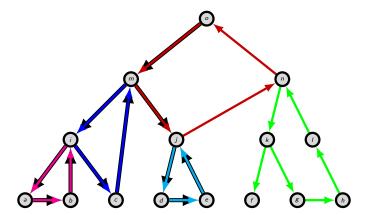


6:28 1.5-Approximation 31/41



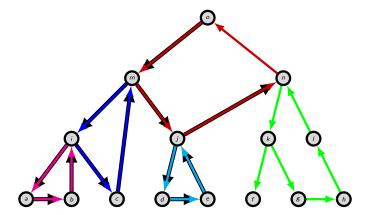


6:28 1.5-Approximation 32/41 Walter Unger 7.10.2024 11:32



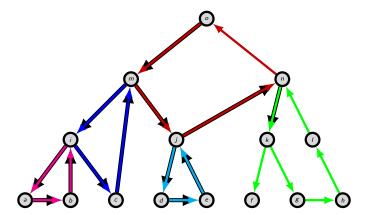


6:28 1.5-Approximation 33/41



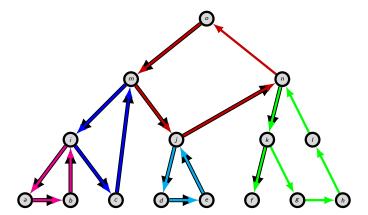


6:28 1.5-Approximation 34/41



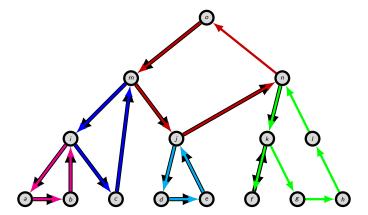


6:28 1.5-Approximation 35/41 Walter Unger 7.10.2024 11:32



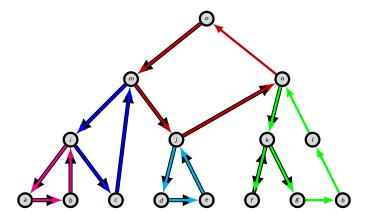


6:28 1.5-Approximation 36/41 Walter Unger 7.10.2024 11:32



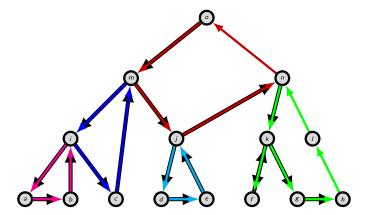


6:28 1.5-Approximation 37/41



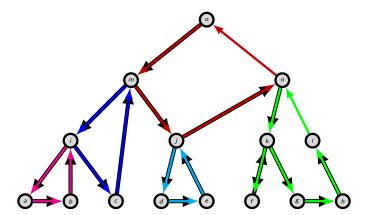


6:28 1.5-Approximation 38/41 Walter Unger 7.10.2024 11:32





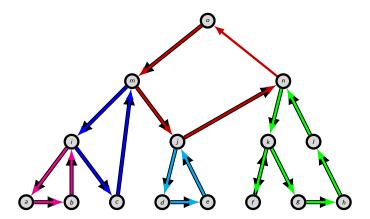
6:28 1.5-Approximation 39/41 Walter Unger 7.10.2024 11:32





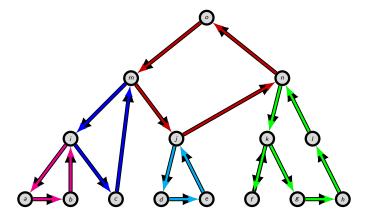
6:28 1.5-Approximation 40/41

Walter Unger 7.10.2024 11:32





6:28 1.5-Approximation 41/41 Walter Unger 7.10.2024 11:32 SS2022





• Gegeben G = (V, E) Eulergraph.

- Gegeben G = (V, E) Eulergraph.

- Gegeben G = (V, E) Eulergraph.
- \odot Wähle Startknoten v_0 .

- Gegeben G = (V, E) Eulergraph.
- \bigcirc Setze i = 0
- Wähle Startknoten v_n.
- **4** Bestimme Kreis C_0 , der bei v_0 startet.

000000000

- Gegeben G = (V, E) Eulergraph.
- \bigcirc Setze i = 0
- Wähle Startknoten v_n.
- **4** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:

- Gegeben G = (V, E) Eulergraph.
 - \bigcirc Setze i = 0
- Wähle Startknoten vo.
- **9** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **9** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.

- Gegeben G = (V, E) Eulergraph.
- ② Setze i = 0
- \odot Wähle Startknoten v_0 .
- **3** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **9** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - **9** Setze i = i + 1.

- Gegeben G = (V, E) Eulergraph.
- \bigcirc Setze i = 0
- Wähle Startknoten vo.
- **a** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **3** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - \bigcirc Setze i = i + 1.
 - **8** Bestimme Kreis C_i , der bei v_i startet.

- Gegeben G = (V, E) Eulergraph.
 - \bigcirc Setze i = 0
- Wähle Startknoten vo.
- **a** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **3** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - \bigcirc Setze i = i + 1.
 - **8** Bestimme Kreis C_i , der bei v_i startet.
- Nun können die Kreise einfach kombiniert werden:

- Gegeben G = (V, E) Eulergraph.
- Setze *i* = 0
- Wähle Startknoten vo.
- **a** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **3** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - \bigcirc Setze i = i + 1.
 - **8** Bestimme Kreis C_i , der bei v_i startet.
- Nun können die Kreise einfach kombiniert werden:
 - Gehe auf C_0 von v_0 bis v_1 .

Einleitung

- Gegeben G = (V, E) Eulergraph.
- \odot Wähle Startknoten v_0 .
- **4** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **1** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - **2** Setze i = i + 1.
 - \odot Bestimme Kreis C_i , der bei v_i startet.
- O Nun können die Kreise einfach kombiniert werden:
 - Gehe auf C_0 von v_0 bis v_1 .
 - ② Durchlaufe rekursiv alle Kreise C_1, C_2, \ldots

- Gegeben G = (V, E) Eulergraph.
 - Setze *i* = 0
- Wähle Startknoten v₀.
- **a** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **9** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - \bigcirc Setze i = i + 1.
 - **8** Bestimme Kreis C_i , der bei v_i startet.
- Nun können die Kreise einfach kombiniert werden:
 - Gehe auf C_0 von v_0 bis v_1 .
 - ② Durchlaufe rekursiv alle Kreise C_1, C_2, \ldots
 - **6** Gehe auf C_0 von v_1 bis v_0 .

Bestimmung des Eulerkreises

- Gegeben G = (V, E) Eulergraph.
 - Setze *i* = 0
- Wähle Startknoten v₀.
- **4** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **9** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - \bigcirc Setze i = i + 1.
 - **8** Bestimme Kreis C_i , der bei v_i startet.
- Nun können die Kreise einfach kombiniert werden:
 - **1** Gehe auf C_0 von v_0 bis v_1 .
 - **2** Durchlaufe rekursiv alle Kreise C_1, C_2, \ldots
 - **6** Gehe auf C_0 von v_1 bis v_0 .

Laufzeit: O(|E|).

- Gegeben G = (V, E) Eulergraph.
 - Setze *i* = 0
- Wähle Startknoten v₀.
- **4** Bestimme Kreis C_0 , der bei v_0 startet.
- Solange es noch ungenutzte Kanten gibt:
 - **9** Bestimme von v_i aus auf C_i den ersten Knoten v_{i+1} mit ungenutzter Kante.
 - \bigcirc Setze i = i + 1.
 - **8** Bestimme Kreis C_i , der bei v_i startet.
- Nun können die Kreise einfach kombiniert werden:
 - **1** Gehe auf C_0 von v_0 bis v_1 .
 - **2** Durchlaufe rekursiv alle Kreise C_1, C_2, \ldots
 - **6** Gehe auf C_0 von v_1 bis v_0 .

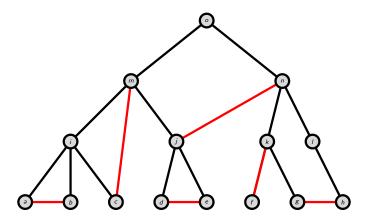
Laufzeit: O(|E|).

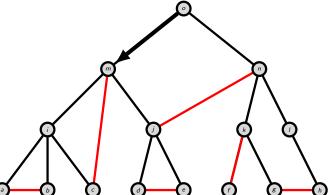


6:30 1.5-Approximation 1/16

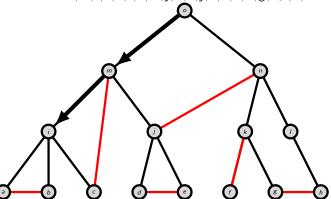
Walter Unger 7.10.2024 11:32

Beispiel (1.5 Approximation)





Beispiel (1.5 Approximation)

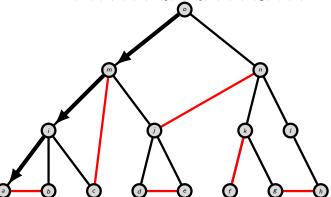


1.5-Approximation 4/16

Einleitung

Walter Unger 7.10.2024 11:32

Beispiel (1.5 Approximation)

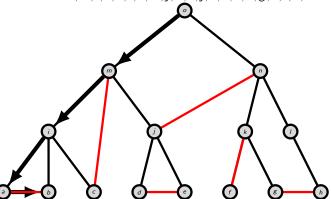


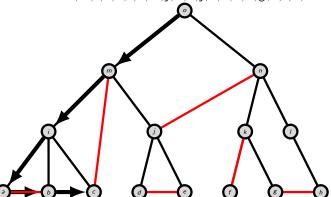


1.5-Approximation 5/16

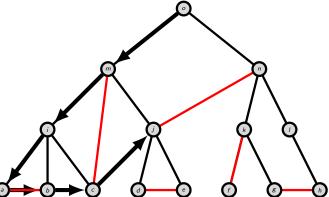
Walter Unger 7.10.2024 11:32

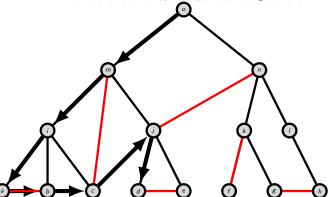
Beispiel (1.5 Approximation)





Beispiel (1.5 Approximation)

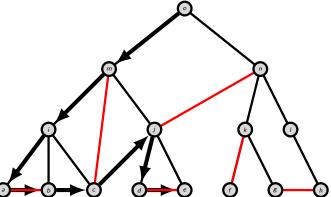


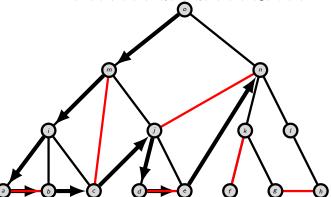


1.5-Approximation 9/16

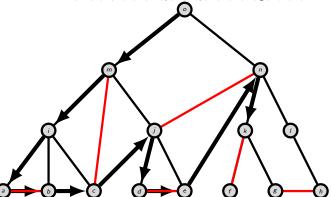
Walter Unger 7.10.2024 11:32

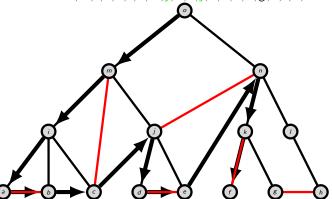
Beispiel (1.5 Approximation)

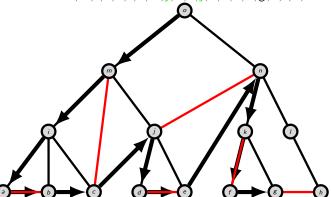




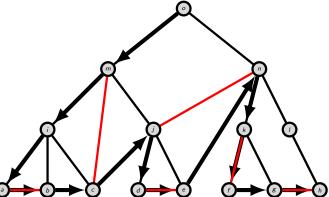
Beispiel (1.5 Approximation)



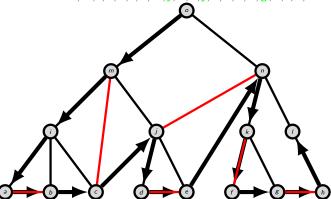


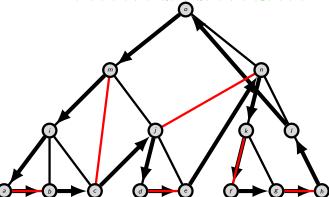


Beispiel (1.5 Approximation)











1.5-Approximation (Beweis)

 Seien T minimaler Spannbaum von G, M das Matching, C' Eulerkreis in T' und C gefundene Lösung.



6:31 1.5-Approximation 2/11

1.5-Approximation (Beweis)

- Seien T minimaler Spannbaum von G, M das Matching, C' Eulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.



000000000

1.5-Approximation (Beweis)

- Seien T minimaler Spannbaum von G, M das Matching, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.

Färbung

000000

SS2022

000000000

- Seien T minimaler Spannbaum von G, M das Matching, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum

Färbung

Einleitung

000000000

- Seien T minimaler Spannbaum von G, M das Matching, C' Eulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum
- $c(M) \leq c(C^*)/2$.

000000000

- Seien T minimaler Spannbaum von G,
 M das Matching,
 C' Eulerkreis in T' und
 C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum
- $c(M) \leq c(C^*)/2$.
 - Siehe dazu nächste Folie.

00000000000

Einleitung

000000000

- Seien T minimaler Spannbaum von G, M das Matching, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum
- $c(M) \leq c(C^*)/2$.
 - Siehe dazu nächste Folie.
- $\frac{c(C)}{c(C^*)}$

000000000

Zentrumsproblem

00000000000

- Seien T minimaler Spannbaum von G, M das Matching, C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum
- $c(M) \leq c(C^*)/2$.
 - Siehe dazu nächste Folie.
- $\frac{c(C)}{c(C^*)} \leqslant \frac{c(T)+c(M)}{c(C^*)}$

Zentrumsproblem

00000000000

- Seien T minimaler Spannbaum von G, M das Matching. C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leqslant c(C^* e) \leqslant c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum
- $c(M) \leq c(C^*)/2$.
 - Siehe dazu nächste Folie.
- $\frac{c(C)}{c(C^*)} \leqslant \frac{c(T) + c(M)}{c(C^*)} \leqslant \frac{c(C^*) + c(C^*)/2}{c(C^*)}$

- Seien T minimaler Spannbaum von G, M das Matching. C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leq c(C^* e) \leq c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum
- $c(M) \leq c(C^*)/2$.
 - Siehe dazu nächste Folie.

$$\bullet \ \frac{c(\mathcal{C})}{c(\mathcal{C}^*)} \leqslant \frac{c(\mathcal{T}) + c(\mathcal{M})}{c(\mathcal{C}^*)} \leqslant \frac{c(\mathcal{C}^*) + c(\mathcal{C}^*)/2}{c(\mathcal{C}^*)} \leqslant \frac{1.5 \cdot c(\mathcal{C}^*)}{c(\mathcal{C}^*)} = 1.5$$

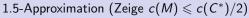
1.5-Approximation (Beweis)

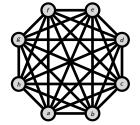
6:31 1.5-Approximation 11/11

Einleitung

- Seien T minimaler Spannbaum von G, M das Matching. C' Fulerkreis in T' und C gefundene Lösung.
- Sei C* eine minimale TSP Tour.
- $c(T) \leq c(C^* e) \leq c(C^*)$.
 - Beachte: $C^* e$ ist ein Spannbaum
- $c(M) \leq c(C^*)/2$.
 - Siehe dazu nächste Folie.
- $\frac{c(C)}{c(C^*)} \leqslant \frac{c(T) + c(M)}{c(C^*)} \leqslant \frac{c(C^*) + c(C^*)/2}{c(C^*)} \leqslant \frac{1.5 \cdot c(C^*)}{c(C^*)} = 1.5$

6:32 1.5-Approximation 1/10



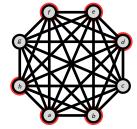




6:32 1.5-Approximation 2/10

Walter Unger 7.10.2024 11:32 SS2022

1.5-Approximation (Zeige $c(M) \leq c(C^*)/2$)



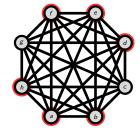
• Sei C^* eine optimale Lösung.



6:32 1.5-Approximation 3/10

Walter Unger 7.10.2024 11:32 SS2022

1.5-Approximation (Zeige $c(M) \leq c(C^*)/2$)

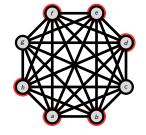


- Sei C* eine optimale Lösung.
- ullet Seien U die ungeraden Knoten auf T.

6:32 1.5-Approximation 4/10

Walter Unger 7.10.2024 11:32 SS2022

1.5-Approximation (Zeige $c(M) \leqslant c(C^*)/2$)

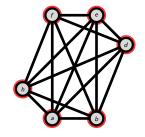


- Sei C* eine optimale Lösung.
- ullet Seien U die ungeraden Knoten auf T.
- Sei C'' der Kreis, der sich aus C^* ergibt durch Überspringen der Knoten $V \setminus U$.



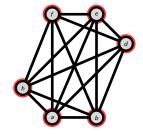
000000000

1.5-Approximation (Zeige $c(M) \leq c(C^*)/2$)

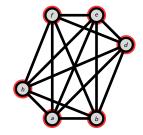


- Sei C* eine optimale Lösung.
- Seien *U* die ungeraden Knoten auf *T*.
- Sei C" der Kreis, der sich aus C* ergibt durch Überspringen der Knoten $V \setminus U$.
- C" hat gerade Länge.

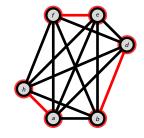




- Sei C* eine optimale Lösung.
- Seien *U* die ungeraden Knoten auf *T*.
- Sei C" der Kreis, der sich aus C* ergibt durch Überspringen der Knoten $V \setminus U$.
- C" hat gerade Länge.
- C'' hat zwei disjunkte perfekte Matchings M_1, M_2

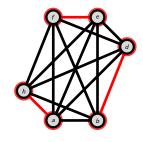


- Sei C* eine optimale Lösung.
- Seien U die ungeraden Knoten auf T.
- Sei C" der Kreis, der sich aus C* ergibt durch Überspringen der Knoten $V \setminus U$.
- C" hat gerade Länge.
- C'' hat zwei disjunkte perfekte Matchings M_1, M_2
- $c(M) \leqslant c(M_1)$ und $c(M) \leqslant c(M_2)$.



- Sei C* eine optimale Lösung.
- Seien U die ungeraden Knoten auf T.
- Sei C" der Kreis, der sich aus C* ergibt durch Überspringen der Knoten $V \setminus U$.
- C" hat gerade Länge.
- C'' hat zwei disjunkte perfekte Matchings M_1, M_2
- $c(M) \leqslant c(M_1)$ und $c(M) \leqslant c(M_2)$.
- $2 \cdot c(M) \leqslant c(M_1) + c(M_2) = c(C'') \leqslant c(C^*)$

1.5-Approximation (Zeige $c(M) \leq c(C^*)/2$)

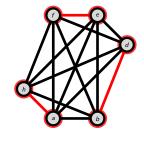


6:32 1.5-Approximation 9/10

Einleitung

- Sei C* eine optimale Lösung.
- Seien U die ungeraden Knoten auf T.
- Sei C" der Kreis, der sich aus C* ergibt durch Überspringen der Knoten $V \setminus U$.
- C" hat gerade Länge.
- C'' hat zwei disjunkte perfekte Matchings M_1, M_2
- $c(M) \leqslant c(M_1)$ und $c(M) \leqslant c(M_2)$.
- $2 \cdot c(M) \leqslant c(M_1) + c(M_2) = c(C'') \leqslant c(C^*)$
- $c(M) \leq c(C^*)/2$

1.5-Approximation (Zeige $c(M) \leq c(C^*)/2$)



- Sei C* eine optimale Lösung.
- Seien U die ungeraden Knoten auf T.
- Sei C" der Kreis, der sich aus C* ergibt durch Überspringen der Knoten $V \setminus U$.
- C" hat gerade Länge.
- C'' hat zwei disjunkte perfekte Matchings M_1, M_2
- $c(M) \leqslant c(M_1)$ und $c(M) \leqslant c(M_2)$.
- $2 \cdot c(M) \leqslant c(M_1) + c(M_2) = c(C'') \leqslant c(C^*)$
- $c(M) \leq c(C^*)/2$

• Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.

- Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.
- Sei $Z \subset V$

- Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.
- Sei $Z \subset V$
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$

- Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.
- Sei Z ⊂ V
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} dist(v, Z)$

- Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.
- Sei Z ⊂ V
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} dist(v, Z)$

• Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.

- Sei Z ⊂ V
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} dist(v, Z)$

Definition (Zentrumsproblem (Entscheidungsvariante))

Gegeben $G = (V, E), k \in \mathbb{N}, L \in \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$. Bestimme $\exists Z : |Z| = k \text{ und } rad(Z) \leq L$.



• Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.

- Sei Z ⊂ V
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} dist(v, Z)$

Definition (Zentrumsproblem (Entscheidungsvariante))

Gegeben $G = (V, E), k \in \mathbb{N}, L \in \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$. Bestimme $\exists Z : |Z| = k \text{ und } rad(Z) \leq L$.

- Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.
- Sei $Z \subset V$
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} dist(v, Z)$

Definition (Zentrumsproblem (Entscheidungsvariante))

Gegeben G = (V, E), $k \in \mathbb{N}$, $L \in \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$.

Bestimme $\exists Z : |Z| = k \text{ und } rad(Z) \leqslant L$.

Definition (Zentrumsproblem)

Gegeben G = (V, E), $k \in \mathbb{N}$ und $c : E \mapsto \mathbb{Q}^+$.

Bestimme Z mit |Z| = k und rad(Z) minimal.



- Gegeben G = (V, E) und $c : E \mapsto \mathbb{Q}^+$.
- Sei Z ⊂ V
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} dist(v, Z)$

Definition (Zentrumsproblem (Entscheidungsvariante))

Gegeben $G = (V, E), k \in \mathbb{N}, L \in \mathbb{Q}^+ \text{ und } c : E \mapsto \mathbb{Q}^+.$

Bestimme $\exists Z : |Z| = k \text{ und } rad(Z) \leq L$.

Definition (Zentrumsproblem)

Gegeben $G = (V, E), k \in \mathbb{N}$ und $c : E \mapsto \mathbb{Q}^+$.

Bestimme Z mit |Z| = k und rad(Z) minimal.

Einleitung Vertex Cover TSP und Delta-TSP Steiner-Bäume Zentrumsproblem Färbung 000000000 00000000 0000000000000 00000000000 00000000000 000000 6:46 Komplexität Walter Unger 7.10.2024 11:32 1/10 SS2022

Komplexität

Theorem

Falls $P \neq NP$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

 Einleitung
 Vertex Cover 000000000
 TSP und Delta-TSP 0000000000
 Steiner-Bäume 0000000000
 Zentrumsproblem 0000000000
 Färbung 00000000000

6:46 Komplexität 2/10 Walter Unger 7:10:2024 11:32 \$\$2022

Komplexität

Theorem

Falls $\mathcal{P} \neq \mathcal{NP}$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

 Einleitung
 Vertex Cover 00000000
 TSP und Delta-TSP 0000000000
 Steiner-Bäume 0000000000
 Zentrumsproblem 0000000000
 Färbung 00000000000

6 Komplexität 3/10 Walter Unger 7.10.2024 11:32 SS2022

Komplexität

Theorem

Falls $\mathcal{P} \neq \mathcal{NP}$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

 Einleitung
 Vertex Cover 00000000
 TSP und Delta-TSP 0000000000
 Steiner-Bäume 0000000000
 Zentrumsproblem 0000000000
 Färbung 00000000000

Komplexität 4/10 Walter Unger 7.10.202411:32 SS2022

Komplexität

Theorem

Falls $\mathcal{P} \neq \mathcal{NP}$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

Theorem

Falls $\mathcal{P} \neq \mathcal{NP}$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

Definition (Dominating Set (Entscheidungsvariante))

Gegeben G = (V, E), $k \in \mathbb{N}$.

Bestimme $\exists D : |D| = k \text{ und } V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}.$

Theorem

Falls $P \neq NP$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

Definition (Dominating Set (Entscheidungsvariante))

Gegeben
$$G = (V, E)$$
, $k \in \mathbb{N}$.
Bestimme $\exists D : |D| = k$ und $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$.

Alternativ:

Theorem

Falls $P \neq NP$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

Definition (Dominating Set (Entscheidungsvariante))

Gegeben
$$G = (V, E)$$
, $k \in \mathbb{N}$.
Bestimme $\exists D : |D| = k$ und $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$.

- Alternativ:
 - $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$ genau dann, wenn

Theorem

Falls $P \neq NP$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

Definition (Dominating Set (Entscheidungsvariante))

Gegeben
$$G = (V, E)$$
, $k \in \mathbb{N}$.
Bestimme $\exists D : |D| = k$ und $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$.

- Alternativ:
 - $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$ genau dann, wenn
 - rad(D) = 1.

Theorem

Falls $\mathcal{P} \neq \mathcal{NP}$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

Definition (Dominating Set (Entscheidungsvariante))

Gegeben
$$G = (V, E)$$
, $k \in \mathbb{N}$.
Bestimme $\exists D : |D| = k$ und $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$.

- Alternativ:
 - $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$ genau dann, wenn
 - rad(D) = 1.

Komplexität

Theorem

Falls $\mathcal{P} \neq \mathcal{NP}$ gilt, gibt es für beliebiges r < 2 keinen Polynomzeitalgorithmus mit Approximationsfaktor r für das Zentrumsproblem.

Beweis:

Reduktion vom Dominating Set Problem.

Definition (Dominating Set (Entscheidungsvariante))

Gegeben
$$G = (V, E)$$
, $k \in \mathbb{N}$.
Bestimme $\exists D : |D| = k$ und $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$.

- Alternativ:
 - $V = D \cup \{v \mid \exists w \in D : \{v, w\} \in E\}$ genau dann, wenn
 - rad(D) = 1.

Reduktion vom Dominating Set Problem.



- Reduktion vom Dominating Set Problem.
- ullet Sei (G, k) Eingabe für das Dominating Set Problem.



- Reduktion vom Dominating Set Problem.
- ullet Sei (G, k) Eingabe für das Dominating Set Problem.
- Setze c(e) = 1 für $e \in E(G)$.

- Reduktion vom Dominating Set Problem.
- Sei (G, k) Eingabe für das Dominating Set Problem.
- Setze c(e) = 1 für $e \in E(G)$.
- Dann hat G Dominating Set der Größe k, falls G Zentrum mit k Knoten und Radius 1 hat.

- Reduktion vom Dominating Set Problem.
- Sei (G, k) Eingabe für das Dominating Set Problem.
- Setze c(e) = 1 für $e \in E(G)$.
- Dann hat G Dominating Set der Größe k, falls G Zentrum mit k Knoten und Radius 1 hat.
- Falls es einen Algorithmus A mit Approximationsfaktor r < 2 gibt,

• Reduktion vom Dominating Set Problem.

- Treduction voin Dominating Set Froblem
- Sei (G, k) Eingabe für das Dominating Set Problem.
- Setze c(e) = 1 für $e \in E(G)$.
- Dann hat G Dominating Set der Größe k, falls G Zentrum mit k Knoten und Radius 1 hat.
- Falls es einen Algorithmus A mit Approximationsfaktor r < 2 gibt,
- So liefert A bei Eingabe (G, 1, k, c) ein Dominating Set Problem der Größe k.

- Reduktion vom Dominating Set Problem.
- Sei (G, k) Eingabe für das Dominating Set Problem.
- Setze c(e) = 1 für $e \in E(G)$.
- Dann hat G Dominating Set der Größe k, falls G Zentrum mit k Knoten und Radius 1 hat.
- Falls es einen Algorithmus A mit Approximationsfaktor r < 2 gibt,
- So liefert A bei Eingabe (G, 1, k, c) ein Dominating Set Problem der Größe k.



• Gegeben G = (V, E), $w : V \mapsto \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$.

- Gegeben G = (V, E), $w : V \mapsto \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$.
- Sei *Z* ⊂ *V*

Zentrumsproblem (mit Knotenkosten)

- Gegeben $G = (V, E), w : V \mapsto \mathbb{Q}^+ \text{ und } c : E \mapsto \mathbb{Q}^+.$
- Sei Z ⊂ V
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$

Einleitung

000000000

- Gegeben G = (V, E), $w : V \mapsto \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$.
- Sei $Z \subset V$
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} w(v) \cdot dist(v, Z)$ von nun an.

- Gegeben G = (V, E), $w : V \mapsto \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$.
- Sei $Z \subset V$
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} w(v) \cdot dist(v, Z)$ von nun an.

Bestimme Z mit |Z| = k und rad(Z) minimal.

Definition (Zentrumsproblem)

Gegeben G = (V, E), $k \in \mathbb{N}$, $w : V \mapsto \mathbb{Q}^+$ und $c : E \mapsto \mathbb{Q}^+$.

Zentrumsproblem (mit Knotenkosten)

- Gegeben $G = (V, E), w : V \mapsto \mathbb{Q}^+ \text{ und } c : E \mapsto \mathbb{Q}^+.$
- Sei Z ⊂ V
- $dist(v, Z) := \min_{z \in Z} dist(z, v)$
- $rad(Z) := max_{v \in V} w(v) \cdot dist(v, Z)$ von nun an.

Definition (Zentrumsproblem)

Gegeben $G = (V, E), k \in \mathbb{N}, w : V \mapsto \mathbb{Q}^+ \text{ und } c : E \mapsto \mathbb{Q}^+.$ Bestimme Z mit |Z| = k und rad(Z) minimal.

Idee

• Wir kennen die Knoten aus Z nicht.

- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.

6:49 Approximation 3/9

- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.
- Idee: suche Z mit Greedy Verfahren für ein festes R.

Walter Unger 7.10.2024 11:32 SS2022

- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.
- Idee: suche Z mit Greedy Verfahren für ein festes R.
 - Solange noch Knoten nicht durch Z überdeckt sind.

Walter Unger 7.10.2024 11:32 SS2022

- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.
- Idee: suche Z mit Greedy Verfahren für ein festes R.
 - Solange noch Knoten nicht durch Z überdeckt sind.
 - Wähle mit Greedy einen weiteren Knoten z zu Z hinzu.

6:49 Approximation 6/9 Walter Unger 7.10.2024 11:32 SS2022

- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.
- Idee: suche Z mit Greedy Verfahren für ein festes R.
 - Solange noch Knoten nicht durch Z überdeckt sind.
 - Wähle mit Greedy einen weiteren Knoten z zu Z hinzu.
 - Alle Knoten im Abstand $2 \cdot R$ werden nun als überdeckt betrachtet.



- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.
- Idee: suche Z mit Greedy Verfahren für ein festes R.
 - Solange noch Knoten nicht durch Z überdeckt sind.
 - Wähle mit Greedy einen weiteren Knoten z zu Z hinzu.
 - Alle Knoten im Abstand $2 \cdot R$ werden nun als überdeckt betrachtet.
- Achtung: der Algorithmus beinhaltet den Approximationsfaktor!

- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.
- Idee: suche Z mit Greedy Verfahren für ein festes R.
 - Solange noch Knoten nicht durch Z überdeckt sind.
 - Wähle mit Greedy einen weiteren Knoten z zu Z hinzu.
 - Alle Knoten im Abstand 2 · R werden nun als überdeckt betrachtet.
- Achtung: der Algorithmus beinhaltet den Approximationsfaktor!
- Teste nun verschiedene Werte R mit Hilfe dieses Greedyverfahrens.

- Wir kennen die Knoten aus Z nicht.
- Wir kennen das Optimum R = rad(Z) nicht.
- Idee: suche Z mit Greedy Verfahren für ein festes R.
 - Solange noch Knoten nicht durch Z überdeckt sind.
 - Wähle mit Greedy einen weiteren Knoten z zu Z hinzu.
 - Alle Knoten im Abstand 2 · R werden nun als überdeckt betrachtet.
- Achtung: der Algorithmus beinhaltet den Approximationsfaktor!
- Teste nun verschiedene Werte R mit Hilfe dieses Greedyverfahrens.

 $\operatorname{argmax}\{X\} = x \text{ mit } c(x) = \max\{c(x') \mid x' \in X\}$

Algorithmus GreedyZentrum(G, c, w, R):

 $\operatorname{argmax}\{X\} = x \text{ mit } c(x) = \max\{c(x') \mid x' \in X\}$

- Algorithmus *GreedyZentrum*(*G*, *c*, *w*, *R*):

 $\operatorname{argmax}\{X\} = x \text{ mit } c(x) = \max\{c(x') \mid x' \in X\}$

- Algorithmus GreedyZentrum(G, c, w, R):

 - Solange $U \neq \emptyset$ mache

$$\operatorname{argmax}\{X\} = x \text{ mit } c(x) = \max\{c(x') \mid x' \in X\}$$

• Algorithmus *GreedyZentrum*(*G*, *c*, *w*, *R*):

- Solange $U \neq \emptyset$ mache

Algorithmus *GreedyZentrum*(*G*, *c*, *w*, *R*):

- $2 := \emptyset und U := V$
- Solange $U \neq \emptyset$ mache
 - $z := \operatorname{argmax}\{w(u) \mid u \in U\}$
 - 2 $Z := Z \cup \{z\}$

$$\operatorname{argmax}\{X\} = x \operatorname{mit} c(x) = \operatorname{max}\{c(x') \mid x' \in X\}$$

- Algorithmus *GreedyZentrum*(*G*, *c*, *w*, *R*):
 - \bigcirc $Z := \emptyset$ und U := V
 - \bullet Solange $U \neq \emptyset$ mache

 - $Z := \overline{Z} \cup \{z\}$
 - $U := U \setminus \{v \in U \mid w(v) \cdot dist(v, z) \leq 2 \cdot R\}$

$$\operatorname{argmax}\{X\} = x \text{ mit } c(x) = \max\{c(x') \mid x' \in X\}$$

- Algorithmus *GreedyZentrum*(*G*, *c*, *w*, *R*):
 - \bigcirc $Z := \emptyset$ und U := V
 - \bigcirc Solange $U \neq \emptyset$ mache

 - $Z := \overline{Z} \cup \{z\}$
 - $U := U \setminus \{v \in U \mid w(v) \cdot dist(v, z) \leq 2 \cdot R\}$
 - Ausgabe: Z

AL 31 6 17 1 (6 5

- Algorithmus GreedyZentrum(G, c, w, R):
 - $2 := \emptyset \text{ und } U := V$ Solange $U \neq \emptyset$ mache
 - Solange $U \neq \emptyset$ mache
 - $z := \operatorname{argmax}\{w(u) \mid u \in U\}$ $Z := Z \cup \{z\}$
 - $U := U \setminus \{v \in U \mid w(v) \cdot dist(v, z) \leq 2 \cdot R\}$
 - Ausgabe: Z
- Sei R^* optimale Lösung mit Z^* . Falls $R \geqslant R^*$, dann liefert GreedyZentrum ein Z mit $|Z| \leqslant |Z^*|$.

- Algorithmus GreedyZentrum(G, c, w, R):
 - $Q := \emptyset \text{ und } U := V$
 - 2 Solange $U \neq \emptyset$ mache

 - **2** $Z := Z \cup \{z\}$
 - Ausgabe: Z
- Sei R^* optimale Lösung mit Z^* . Falls $R \geqslant R^*$, dann liefert GreedyZentrum ein Z mit $|Z| \leqslant |Z^*|$.
- Im Hauptalgorithmus wird GreedyZentrum daher für alle interessanten Werte für R aufgerufen.

Algorithmus GreedyZentrum(G, c, w, R):

- - \bigcirc Solange $U \neq \emptyset$ mache
 - $\underline{\mathbf{z}} := \underset{\mathbf{z}}{\operatorname{argmax}} \{ w(u) \mid u \in U \}$
 - **2** $Z := Z \cup \{z\}$
 - Ausgabe: Z
- Sei R^* optimale Lösung mit Z^* . Falls $R \geqslant R^*$, dann liefert GreedyZentrum ein Z mit $|Z| \leqslant |Z^*|$.
- Im Hauptalgorithmus wird GreedyZentrum daher für alle interessanten Werte für R aufgerufen.
- Definiere: f(R) := |GreedyZentrum(G, c, w, R)|

- Algorithmus GreedyZentrum(G, c, w, R):
 - $Q := \emptyset$ und U := V
 - 2 Solange $U \neq \emptyset$ mache
 - - **2** $Z := Z \cup \{z\}$

 - $U := U \setminus \{v \in U \mid w(v) \cdot dist(v, z) \leq 2 \cdot R\}$
 - Ausgabe: Z
- Sei R^* optimale Lösung mit Z^* . Falls $R \ge R^*$, dann liefert *GreedyZentrum* ein Z mit $|Z| \leq |Z^*|$.
- Im Hauptalgorithmus wird GreedyZentrum daher für alle interessanten Werte für R aufgerufen.
- Definiere: f(R) := |GreedyZentrum(G, c, w, R)|
 - Beachte f(R) ist aber nicht monoton.

- Algorithmus GreedyZentrum(G, c, w, R):
 - $Q := \emptyset$ und U := V
 - Solange $U \neq \emptyset$ mache
 - $z := \operatorname{argmax}\{w(u) \mid u \in U\}$
 - $Z := Z \cup \{z\}$
 - $U := U \setminus \{ v \in U \mid w(v) \cdot dist(v, z) \leq 2 \cdot R \}$
 - Ausgabe: Z
 - Sei R* optimale Lösung mit Z*. Falls R ≥ R*, dann liefert GreedyZentrum ein Z mit |Z| ≤ |Z*|.
- Im Hauptalgorithmus wird GreedyZentrum daher für alle interessanten Werte für R aufgerufen.
- Definiere: f(R) := |GreedyZentrum(G, c, w, R)|
 - Beachte f(R) ist aber nicht monoton.
- Definiere: $R' = \max_{v \in V} w(v) \cdot \sum_{e \in F} c(e)$

- Algorithmus *GreedyZentrum*(*G*, *c*, *w*, *R*):

 - Solange $U \neq \emptyset$ mache
 - $z := \operatorname{argmax}\{w(u) \mid u \in U\}$
 - $Z := Z \cup \{z\}$
 - $U := U \setminus \{v \in U \mid w(v) \cdot dist(v, z) \leq 2 \cdot R\}$
 - Ausgabe: Z
- Sei R^* optimale Lösung mit Z^* . Falls $R \geqslant R^*$, dann liefert GreedyZentrum ein Z mit $|Z| \leqslant |Z^*|$.
- Im Hauptalgorithmus wird GreedyZentrum daher für alle interessanten Werte für R aufgerufen.
- Definiere: f(R) := |GreedyZentrum(G, c, w, R)|
 - Beachte f(R) ist aber nicht monoton.
- Definiere: $R' = \max_{v \in V} w(v) \cdot \sum_{e \in F} c(e)$
 - Mit so einem großen Radius kann ein Knoten alles überdecken.

Approximation

Algorithmus GreedyZentrum(G, c, w, R):

- $Q := \emptyset \text{ und } U := V$
- $otin{f 2}$ Solange $U \neq \emptyset$ mache

 - $Z := Z \cup \{z\}$
- Ausgabe: Z
- Sei R^* optimale Lösung mit Z^* . Falls $R \geqslant R^*$, dann liefert GreedyZentrum ein Z mit $|Z| \leqslant |Z^*|$.
- Im Hauptalgorithmus wird GreedyZentrum daher für alle interessanten Werte für R aufgerufen.
- Definiere: f(R) := |GreedyZentrum(G, c, w, R)|
 - Beachte f(R) ist aber nicht monoton.
- Definiere: $R' = \max_{v \in V} w(v) \cdot \sum_{e \in F} c(e)$
 - Mit so einem großen Radius kann ein Knoten alles überdecken.
 - f(R') = 1 und f(0) = n

- Algorithmus GreedyZentrum(G, c, w, R):
 - O $Z := \emptyset$ und U := V
 - Solange $U \neq \emptyset$ mache
 - $z := \operatorname{argmax} \{ w(u) \mid u \in U \}$
 - $Z := \bar{Z} \cup \{z\}$
 - Ausgabe: Z
 - Sei R^* optimale Lösung mit Z^* . Falls $R \geqslant R^*$, dann liefert GreedyZentrum ein Z mit $|Z| \leqslant |Z^*|$.
- Im Hauptalgorithmus wird GreedyZentrum daher für alle interessanten Werte für R aufgerufen.
- Definiere: f(R) := |GreedyZentrum(G, c, w, R)|
 - Beachte f(R) ist aber nicht monoton.
- Definiere: $R' = \max_{v \in V} w(v) \cdot \sum_{e \in F} c(e)$
 - Mit so einem großen Radius kann ein Knoten alles überdecken.
 - f(R') = 1 und f(0) = n

Approximation 1/21

Einleitung

000000000

Walter Unger 7.10.2024 11:32



Bildhaftes Beispiel (R = 2)

Einleitung

000000000



Approximation 3/21

Einleitung

000000000

Walter Unger 7.10.2024 11:32 SS2022

Approximation 4/21

Einleitung

Walter Unger 7.10.2024 11:32 SS2022

$$w(00) = w(06) = w(23) = w(90) = w(96) = w(51) = w(55) = 2$$

100 10 20 30 40 60 60 70 85 65

15 25 35 45 65 65 75 85 65

101 12 23 33 43 53 33 73 83 93

102 12 22 32 42 52 62 72 82 62

101 11 21 31 41 51 61 71 81 91



5:51 Approximation 5/21

Einleitung

000000000



:51 Approximation 6/21

Einleitung

000000000



Approximation 7/21

Walter Unger 7.10.2024 11:32 SS2022

Bildhaftes Beispiel (R = 2)

Einleitung

000000000



Approximation 8/21

Walter Unger 7.10.2024 11:32

Bildhaftes Beispiel (R = 2)

Einleitung



:51 Approximation 9/21

Einleitung

000000000



 Approximation
 10/21
 Walter Unger 7.10.2024 11:32
 SS2022



Approximation 11/21

Walter Unger 7.10.2024 11:32

Walter Unger 7.10.2024 11:32 Approximation 12/21 SS2022

Bildhaftes Beispiel (R = 2)

Einleitung





Walter Unger 7.10.2024 11:32 Approximation 14/21



Approximation 15/21

Einleitung

Walter Unger 7.10.2024 11:32 SS2022



Approximation 16/21

Einleitung

000000000

Walter Unger 7.10.2024 11:32 SS2022

Approximation 17/21

Einleitung

Walter Unger 7.10.2024 11:32



Approximation 18/21

Einleitung

000000000

Walter Unger 7.10.2024 11:32



6:51 Approximation 19/21

Einleitung

000000000

Walter Unger 7.10.2024 11:32 SS2022

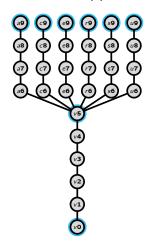
Approximation 20/21 Walter Unger 7.10.2024 11:32 SS2022

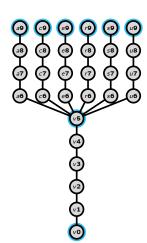


:51 Approximation 21/21

Walter Unger 7.10.2024 11:32 SS2022

Beispiel für f(R) ist nicht monoton



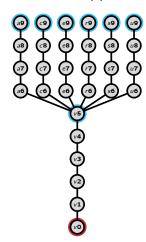


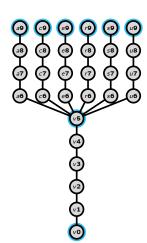


6:52 Approximation 2/25

Walter Unger 7.10.2024 11:32

Beispiel für f(R) ist nicht monoton



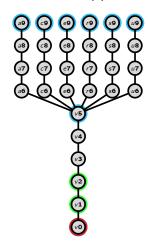


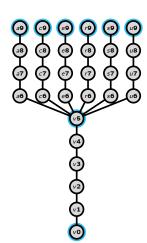


6:52 Approximation 3/25

Walter Unger 7.10.2024 11:32

Beispiel für f(R) ist nicht monoton



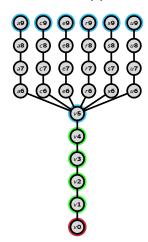


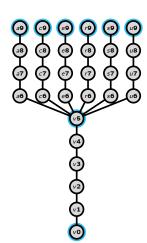


6:52 Approximation 4/25

Walter Unger 7.10.2024 11:32

Beispiel für f(R) ist nicht monoton



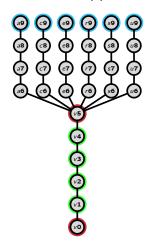


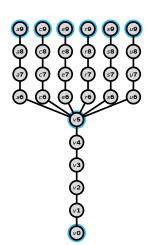


6:52 Approximation 5/25

Walter Unger 7.10.2024 11:32

Beispiel für f(R) ist nicht monoton



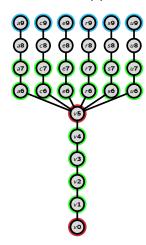


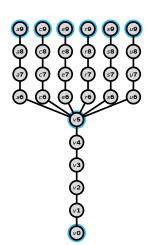


6:52 Approximation 6/25

Walter Unger 7.10.2024 11:32

Beispiel für f(R) ist nicht monoton



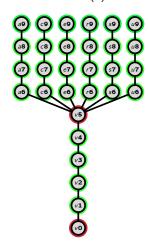


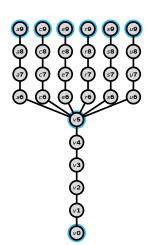


6:52 Approximation 7/25

Walter Unger 7.10.2024 11:32

Beispiel für f(R) ist nicht monoton





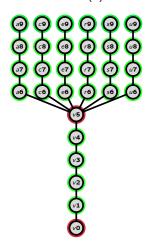


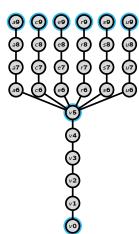
Einleitung

000000000

6:52 Approximation 8/25

Bestimme f(2) = 2



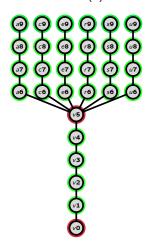


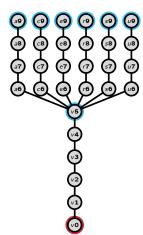


Einleitung

000000000

Bestimme f(2) = 2







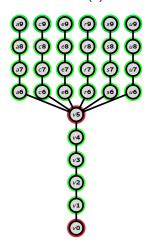
Beispiel für f(R) ist nicht monoton

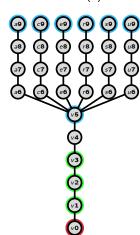
Einleitung

000000000

6:52 Approximation 10/25

Bestimme
$$f(2) = 2$$





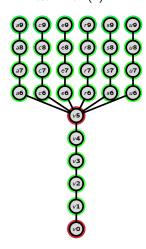


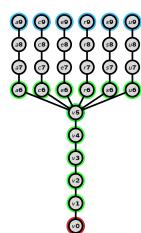
Zentrumsproblem

Einleitung

000000000

Bestimme f(2) = 2





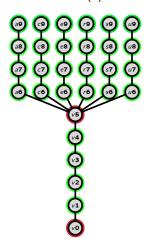


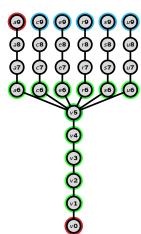
6:52 Approximation 12/25 Beispiel für f(R) ist nicht monoton

Einleitung

000000000

Bestimme f(2) = 2





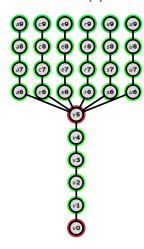


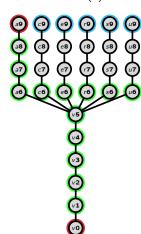
Einleitung

000000000

6:52 Approximation 13/25

Bestimme
$$f(2) = 2$$





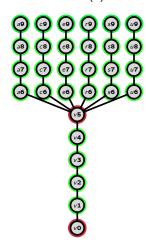


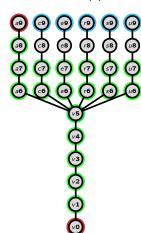
Einleitung

000000000

6:52 Approximation 14/25

Bestimme
$$f(2) = 2$$





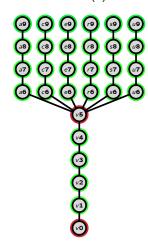


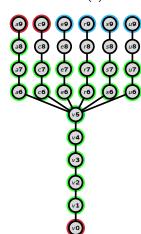
Einleitung

000000000

6:52 Approximation 15/25

Bestimme
$$f(2) = 2$$





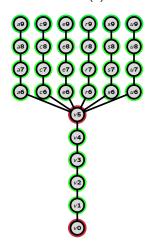


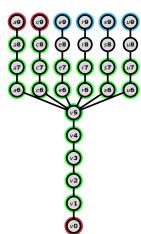
Einleitung

000000000

Beispiel für f(R) ist nicht monoton

Bestimme f(2) = 2





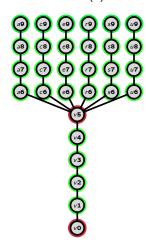


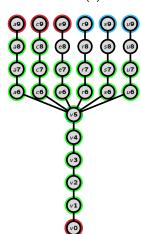
Einleitung

000000000

6:52 Approximation 17/25

Bestimme
$$f(2) = 2$$





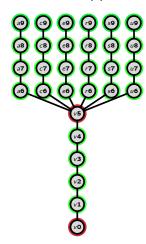


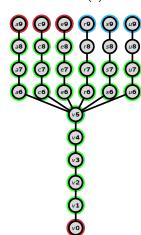
Einleitung

000000000

6:52 Approximation 18/25

Bestimme
$$f(2) = 2$$





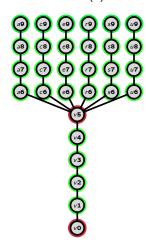


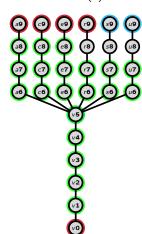
6:52 Approximation 19/25 Beispiel für f(R) ist nicht monoton

Einleitung

000000000

Bestimme
$$f(2) = 2$$





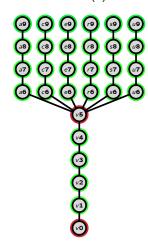


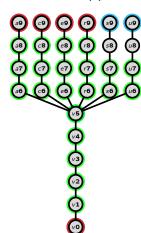
Einleitung

000000000

6:52 Approximation 20/25

Bestimme
$$f(2) = 2$$





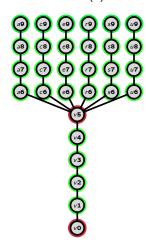


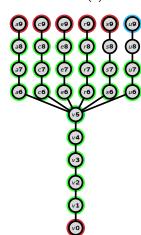
6:52 Approximation 21/25 Beispiel für f(R) ist nicht monoton

Einleitung

000000000

Bestimme
$$f(2) = 2$$



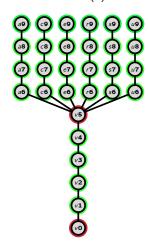


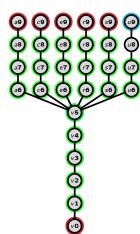


Einleitung

000000000

Bestimme f(2) = 2





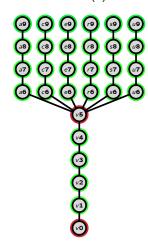


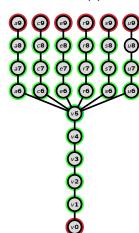
6:52 Approximation 23/25 Beispiel für f(R) ist nicht monoton

Einleitung

000000000

Bestimme
$$f(2) = 2$$





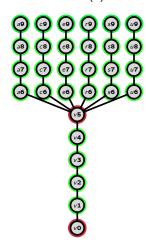


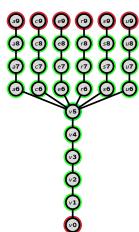
6:52 Approximation 24/25 Beispiel für f(R) ist nicht monoton

Einleitung

000000000

Bestimme f(2) = 2





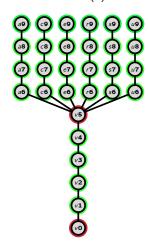


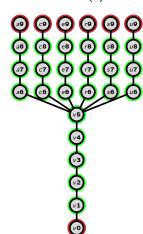
6:52 Approximation 25/25 Beispiel für f(R) ist nicht monoton

Einleitung

000000000

Bestimme
$$f(2) = 2$$







6:53 Approximation 1/16

Approximation

Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*)$.

6:53 Approximation 2/16

Approximation

Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*)$.

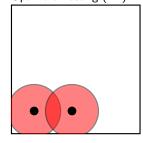
A Approximation 3/10

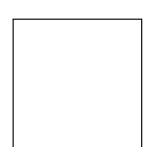
Approximation

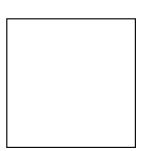
Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \text{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.







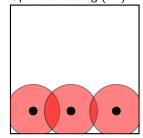
Approximation 4/16

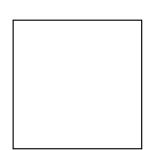
Approximation

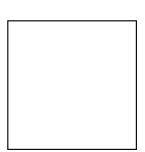
Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*).$

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.







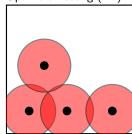
Approximation 5/16

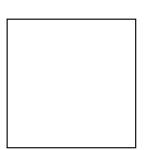
Approximation

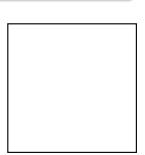
Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*).$

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.



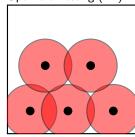


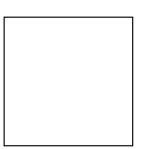


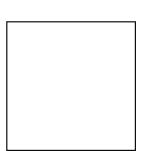
Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \text{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.







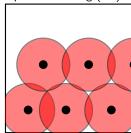
Approximation 7/16

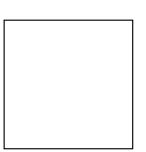
Approximation

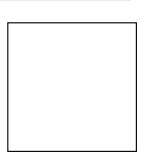
Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.







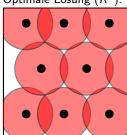
Approximation 8/16

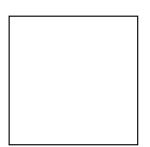
Approximation

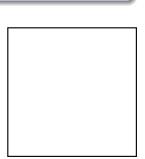
Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.







Approximation 9/16

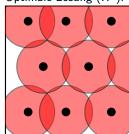
Approximation

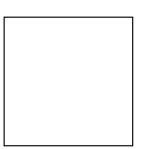
Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*).$

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :





 Einleitung
 Vertex Cover
 TSP und Delta-TSP
 Steiner-Bäume
 Zentrumsproblem
 Färbung

 000000000
 00000000
 00000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 00000000000
 0000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 000000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 000000000000
 000000000000
 00000000000
 000000000000
 00000000000
 00000000000000
 00000000000
 00000000000
 000000000000
 000000000000
 0000000000000
 00000000000
 00000000000
 00000000000
 0000000000
 00000000000
 000000000000
 00000000000
 000000000000000
 00000000000
 000000000000
 000000000000
 000000000000
 00000000000000
 000000000000000
 00000000000000
 00000000000000000
 000000000000000000
 0000000000000000000
 0000000000000000000
 0000000000000000000000000
 00000000000000000000000

Approximation

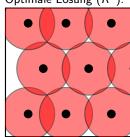
ргожинато

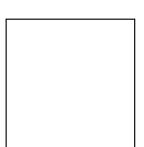
Lemma

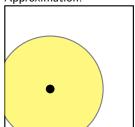
Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \text{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :







 Einleitung
 Vertex Cover
 TSP und Delta-TSP
 Steiner-Bäume
 Zentrumsproblem
 Färbung

 000000000
 00000000
 00000000000
 00000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 00000000000
 0000000000
 0000000000
 00000000000
 0000000000
 0000000000
 0000000000
 0000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 00000000000
 000000000000
 000000000000
 00000000000
 00000000000000
 00000000000
 000000000000
 00000000000
 00000000000
 00000000000
 0000000000
 00000000000
 00000000000
 0000000000
 0000000000
 0000000000
 0000000000
 0000000000
 00000000000
 00000000000
 0000000000
 00000000000
 0000000000
 0000000000
 00000000000
 000000000000
 0000000000000
 000000000000
 000000000000
 0000000000000
 00000000000000
 000000000000000
 000000000000000000
 0000000000000000000
 00000000000000

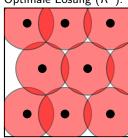
Approximation

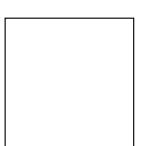
Lemma

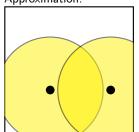
Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \text{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :







Einleitung Vertex Cover TSP und Delta-TSP Steiner-Bäume Zentrumsproblem Färbung 000000000 0000000000000 00000000 0000000000000 00000000000 000000 Approximation 12/16 Walter Unger 7.10.2024 11:32 SS2022

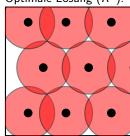
Approximation

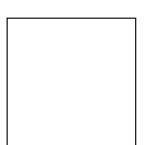
Lemma

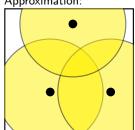
Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*).$

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :







6:53 Approximation 13/16 Walter Unger 7.10.2024 11:32 \$\$2022 Approximation

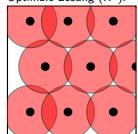
.

Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \text{rad}(Z^*)$.

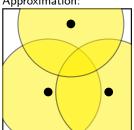
Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :



V 1.1





Approximation 14/16

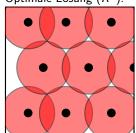
Walter Unger 7.10.2024 11:32 SS2022 Approximation

Lemma

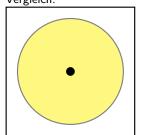
Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*).$

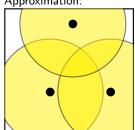
Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :



Vergleich:





Einleitung Vertex Cover TSP und Delta-TSP Steiner-Bäume Zentrumsproblem Färbung 000000000 0000000000000 00000000 0000000000000 00000000000 000000 Approximation 15/16 Walter Unger 7.10.2024 11:32 SS2022

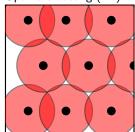
Approximation

Lemma

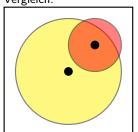
Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*).$

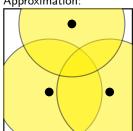
Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :



Vergleich:







Einleitung Vertex Cover TSP und Delta-TSP Steiner-Bäume Zentrumsproblem Färbung 000000000 0000000000000 00000000 0000000000000 00000000000 000000 Approximation 16/16

Approximation

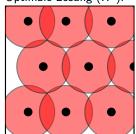
Walter Unger 7.10.2024 11:32 SS2022

Lemma

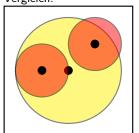
Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalen Radius und $R^* = \operatorname{rad}(Z^*).$

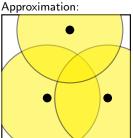
Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

Optimale Lösung (R^*) :



Vergleich:





Approximation 1/11

Approximation

Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \operatorname{rad}(Z^*)$.





54 Approximation 2/11

Walter Unger 7.10.2024 11:32 SS2022

Approximation

Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalem Radius und $R^* = rad(Z^*)$.

Dann gilt: $f(R) \leqslant k$ für alle $R \geqslant R^*$.

• Sei
$$Z^* = \{z_1, z_2, \dots, z_k\}.$$





54 Approximation 3/11

Walter Unger 7.10.2024 11:32 SS2022

Approximation

Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \text{rad}(Z^*)$.

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$



Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \operatorname{rad}(Z^*).$

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.





Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \operatorname{rad}(Z^*).$

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.
- Sei $z \in V_i$ der in der Schleife gewählte Knoten für ein i. Damit gilt:

$$w(v) \cdot dist(v, z) \leqslant w(v) \cdot (dist(v, v_i) + dist(v_i, z))$$



Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \operatorname{rad}(Z^*).$

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.
- Sei $z \in V_i$ der in der Schleife gewählte Knoten für ein i. Damit gilt:

$$w(v) \cdot dist(v, z) \leqslant w(v) \cdot (dist(v, v_i) + dist(v_i, z))$$





Einleitung

000000000

Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \text{rad}(Z^*)$.

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.
- Sei $z \in V_i$ der in der Schleife gewählte Knoten für ein i. Damit gilt:

$$w(v) \cdot dist(v, z) \leqslant w(v) \cdot (dist(v, v_i) + dist(v_i, z)) \leqslant w(v) \cdot dist(v, v_i) + w(z) \cdot dist(v_i, z)$$



Approximation 8/11

Einleitung

000000000

Approximation

Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \operatorname{rad}(Z^*).$

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.
- Sei $z \in V_i$ der in der Schleife gewählte Knoten für ein i. Damit gilt:

$$\begin{array}{lll} w(v) \cdot dist(v,z) & \leqslant & w(v) \cdot (dist(v,v_i) + dist(v_i,z)) \\ & \leqslant & w(v) \cdot dist(v,v_i) + w(z) \cdot dist(v_i,z) \\ & \leqslant & 2 \cdot R^* \end{array}$$



Einleitung

Lemma

Sei $G = (V, E) Z^*$ ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \operatorname{rad}(Z^*).$

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}$.
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.
- Sei $z \in V_i$ der in der Schleife gewählte Knoten für ein i. Damit gilt:

$$\begin{array}{lll} \textit{w}(\textit{v}) \cdot \textit{dist}(\textit{v}, \textit{z}) & \leqslant & \textit{w}(\textit{v}) \cdot (\textit{dist}(\textit{v}, \textit{v}_i) + \textit{dist}(\textit{v}_i, \textit{z})) \\ & \leqslant & \textit{w}(\textit{v}) \cdot \textit{dist}(\textit{v}, \textit{v}_i) + \textit{w}(\textit{z}) \cdot \textit{dist}(\textit{v}_i, \textit{z}) \\ & \leqslant & 2 \cdot \textit{R}^* \\ & \leqslant & 2 \cdot \textit{R} \end{array}$$

• Alle Knoten aus V_i sind am Ende des Schleifendurchlaufes aus U gelöscht.





Einleitung

000000000

Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \text{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.
- Sei $z \in V_i$ der in der Schleife gewählte Knoten für ein i. Damit gilt:

$$\begin{array}{lll} \textit{w(v)} \cdot \textit{dist}(\textit{v},\textit{z}) & \leqslant & \textit{w(v)} \cdot (\textit{dist}(\textit{v},\textit{v}_i) + \textit{dist}(\textit{v}_i,\textit{z})) \\ & \leqslant & \textit{w(v)} \cdot \textit{dist}(\textit{v},\textit{v}_i) + \textit{w(z)} \cdot \textit{dist}(\textit{v}_i,\textit{z}) \\ & \leqslant & 2 \cdot R^* \\ & \leqslant & 2 \cdot R \end{array}$$

- Alle Knoten aus V_i sind am Ende des Schleifendurchlaufes aus U gelöscht.
- Schleife terminiert nach höchstens $k = |Z^*|$ Runden. Und es gilt: $|Z| \leq k$.





Einleitung

000000000

Lemma

Sei G = (V, E) Z^* ein Zentrum aus k Knoten mit minimalem Radius und $R^* = \text{rad}(Z^*)$.

Dann gilt: $f(R) \leq k$ für alle $R \geq R^*$.

- Sei $Z^* = \{z_1, z_2, \dots, z_k\}.$
- Sei $V_i := \{v \in V \mid w(v) \cdot dist(v, z_i) \leqslant R^*\}.$
- Damit $\bigcup_{i=1}^k V_i = V$.
- Sei $z \in V_i$ der in der Schleife gewählte Knoten für ein i. Damit gilt:

$$\begin{array}{lll} \textit{w}(\textit{v}) \cdot \textit{dist}(\textit{v}, \textit{z}) & \leqslant & \textit{w}(\textit{v}) \cdot (\textit{dist}(\textit{v}, \textit{v}_i) + \textit{dist}(\textit{v}_i, \textit{z})) \\ & \leqslant & \textit{w}(\textit{v}) \cdot \textit{dist}(\textit{v}, \textit{v}_i) + \textit{w}(\textit{z}) \cdot \textit{dist}(\textit{v}_i, \textit{z}) \\ & \leqslant & 2 \cdot \textit{R}^* \\ & \leqslant & 2 \cdot \textit{R} \end{array}$$

- Alle Knoten aus V_i sind am Ende des Schleifendurchlaufes aus U gelöscht.
- Schleife terminiert nach höchstens $k = |Z^*|$ Runden. Und es gilt: $|Z| \leq k$.



• Es werden nun die alle möglichen Distanzwerte untersucht:



- Es werden nun die alle möglichen Distanzwerte untersucht:
 - $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$

- Es werden nun die alle möglichen Distanzwerte untersucht:
 - $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$
 - $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \leq n \cdot (n-1)$

- Es werden nun die alle möglichen Distanzwerte untersucht:
 - $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$ • $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \le n \cdot (n-1)$
- Algorithmus *ApproxZentrum*(*G*, *c*, *w*, *k*)

- Es werden nun die alle möglichen Distanzwerte untersucht:
 - $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$ • $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \le n \cdot (n-1)$
- Algorithmus *ApproxZentrum*(*G*, *c*, *w*, *k*)
 - **1** Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).

• Es werden nun die alle möglichen Distanzwerte untersucht:

•
$$\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$$

• $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \le n \cdot (n-1)$

- Algorithmus ApproxZentrum(G, c, w, k)
 - **1** Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).
 - **3** Sortiere Werte $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$.

• Es werden nun die alle möglichen Distanzwerte untersucht:

•
$$\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$$

• $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \le n \cdot (n-1)$

- Algorithmus ApproxZentrum(G, c, w, k)
 - 1 Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).
 - Sortiere Werte $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$.
 - Seien $R_1 < R_2 < \ldots < R_{anz}$ diese Werte.

- Es werden nun die alle möglichen Distanzwerte untersucht:
 - $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$
 - anz := $|\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \leq n \cdot (n-1)$
- Algorithmus ApproxZentrum(G, c, w, k)
 - 1 Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).
 - Sortiere Werte $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$.
 - Seien $R_1 < R_2 < \ldots < R_{anz}$ diese Werte.
 - 0 i := 0, Z = V.

• Es werden nun die alle möglichen Distanzwerte untersucht:

•
$$\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$$

• $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \le n \cdot (n-1)$

- Algorithmus ApproxZentrum(G, c, w, k)
 - **1** Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).
 - **2** Sortiere Werte $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$.
 - Seien $R_1 < R_2 < \ldots < R_{anz}$ diese Werte.
 - 0 i := 0, Z = V.
 - **5** Solange |Z| > k führe aus:

• Es werden nun die alle möglichen Distanzwerte untersucht:

•
$$\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$$

• $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \le n \cdot (n-1)$

- Algorithmus ApproxZentrum(G, c, w, k)
 - 1 Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).
 - Sortiere Werte $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$.
 - Seien $R_1 < R_2 < \ldots < R_{anz}$ diese Werte.
 - i := 0, Z = V.
 - Solange |Z| > k führe aus:
 - 0 i = i + 1.

- Es werden nun die alle möglichen Distanzwerte untersucht:
 - $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$
 - anz := $|\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \leq n \cdot (n-1)$
- Algorithmus ApproxZentrum(G, c, w, k)
 - 1 Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).
 - Sortiere Werte $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$.
 - Seien $R_1 < R_2 < \ldots < R_{anz}$ diese Werte.
 - i := 0, Z = V.
 - **5** Solange |Z| > k führe aus:
 - 0 i = i + 1.
 - $Z := GreedvZentrum(G, c, w, R_i)$

• Es werden nun die alle möglichen Distanzwerte untersucht:

•
$$\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$$

• $anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \le n \cdot (n-1)$

1 Bestimme Distanzmatrix $(dist(u, v))_{u,v \in V}$ für (G, c).

3 Sortiere Werte $\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$.

Seien $R_1 < R_2 < \ldots < R_{anz}$ diese Werte.

4
$$i := 0$$
, $Z = V$.
Solange $|Z| > k$ führe aus:

$$i = i + 1$$
.

$$2 := GreedyZentrum(G, c, w, R_i).$$

Ausgabe: Z.

• Es werden nun die alle möglichen Distanzwerte untersucht:

•
$$\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$$

•
$$anz := |\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}| \leqslant n \cdot (n-1)$$

Algorithmus ApproxZentrum(G, c, w, k)

1 Bestimme Distanzmatrix
$$(dist(u, v))_{u,v \in V}$$
 für (G, c) .

3 Sortiere Werte
$$\{w(u) \cdot dist(u, v) \mid u, v \in V \land u \neq v\}$$
.

Seien
$$R_1 < R_2 < \ldots < R_{anz}$$
 diese Werte.

$$i := 0, Z = V.$$

Solange
$$|Z| > k$$
 führe aus:
 $i = i + 1$.

$$Z := GreedyZentrum(G, c, w, R_i).$$

 Einleitung
 Vertex Cover
 TSP und Delta-TSP
 Steiner-Bäume
 Zentrumsproblem
 Färbung

 0000000000
 000000000
 00000000000
 00000000000
 00000000000
 0000000000

 6:56
 Approximation
 1/12
 Walter Unger 7:10.2024 11:32
 \$\$25202

6:56 Approximation 1/12

Approximation

Theorem

Approximation 2/12

Walter Unger 7.10.2024 11:32 SS2022

Approximation

Theorem

Das Zentrumsproblem ist mit einem Faktor von 2 in Zeit $O(n^4)$ approximierbar.

Wegen dem obigen Lemma terminiert die Schleife.



Approximation 3/12

Walter Unger 7.10.2024 11:32 SS2022

Approximation

Theorem

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei in die Anzahl der Schleifendurchläufe.



6 Approximation 4/12

Walter Unger 7.10.2024 11:32 SS2022

Approximation

Theorem

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei i₀ die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit rad(Z) ≤ 2 · R_{io}

66 Approximation 5/12

Walter Unger 7.10.2024 11:32 SS2022

Approximation

Theorem

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei io die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{i_0}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leqslant 2 \cdot R_{i_0}$

Theorem

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei in die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{in}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leq 2 \cdot R_{in}$
- Damit ist auch Z eine 2 Approximation.

Theorem

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei in die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{in}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leq 2 \cdot R_{in}$
- Damit ist auch Z eine 2 Approximation.
- Laufzeit:

Theorem

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei in die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{in}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leq 2 \cdot R_{in}$
- Damit ist auch Z eine 2 Approximation.
- Laufzeit:
 - Distanzmatrix: O(n³).

Theorem

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei i₀ die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{i_0}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leqslant 2 \cdot R_{i_0}$
- Damit ist auch Z eine 2 Approximation.
- Laufzeit:
 - Distanzmatrix: $O(n^3)$.
 - Sortierung: $O(n^2 \log n)$.

$\mathsf{Theorem}$

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei in die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{in}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leq 2 \cdot R_{in}$
- Damit ist auch Z eine 2 Approximation.
- Laufzeit:
 - Distanzmatrix: O(n³).
 - Sortierung: $O(n^2 \log n)$.
 - GreedyZentrum: O(n²).

$\mathsf{Theorem}$

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei in die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{in}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leq 2 \cdot R_{in}$
- Damit ist auch Z eine 2 Approximation.
- Laufzeit:
 - Distanzmatrix: O(n³).
 - Sortierung: $O(n^2 \log n)$.
 - GreedyZentrum: O(n²).
- Gesamtlaufzeit O(n⁴).

$\mathsf{Theorem}$

- Wegen dem obigen Lemma terminiert die Schleife.
- Sei in die Anzahl der Schleifendurchläufe.
- D.h. der Durchlauf mit Ausgabe Z mit $rad(Z) \leq 2 \cdot R_{in}$
- Durch Konstrukton von *GreedyZentrum* gilt: $rad(Z) \leq 2 \cdot R_{in}$
- Damit ist auch Z eine 2 Approximation.
- Laufzeit:
 - Distanzmatrix: O(n³).
 - Sortierung: $O(n^2 \log n)$.
 - GreedyZentrum: O(n²).
- Gesamtlaufzeit O(n⁴).

6:62 Aussagen 1/7

Literatur

Vazirani: Approximation Algorithms, Springer Verlag, 2001.

Walter Unger 7.10.2024 11:32 SS2022

- Vazirani: Approximation Algorithms, Springer Verlag, 2001.
- Jansen, Margraf: Approximative Algorithmen und Nichtapproximierbarkeit, de Gruyter, 2008.

- Vazirani: Approximation Algorithms, Springer Verlag, 2001.
- Jansen, Margraf: Approximative Algorithmen und Nichtapproximierbarkeit, de Gruyter, 2008.
- Wanka: Approximationsalgorithmen Eine Einführung, Teubner Verlag 2006.



- Vazirani: Approximation Algorithms, Springer Verlag, 2001.
- Jansen, Margraf: Approximative Algorithmen und Nichtapproximierbarkeit, de Gruyter, 2008.
- Wanka: Approximationsalgorithmen Eine Einführung, Teubner Verlag 2006.
- Hochbaum: Approximation Algorithms for NP-Hard Problems, Thomson Publishing, 1996.

- Vazirani: Approximation Algorithms, Springer Verlag, 2001.
- Jansen, Margraf: Approximative Algorithmen und Nichtapproximierbarkeit, de Gruyter, 2008.
- Wanka: Approximationsalgorithmen Eine Einführung, Teubner Verlag 2006.
- Hochbaum: Approximation Algorithms for NP-Hard Problems, Thomson Publishing, 1996.
- Ausiello, Crescenzi, Gambosi, Kann, Marchetti-Spaccamela, Protasi: Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties, Springer Verlag, 1999.

- Vazirani: Approximation Algorithms, Springer Verlag, 2001.
- Jansen, Margraf: Approximative Algorithmen und Nichtapproximierbarkeit, de Gruyter, 2008.
- Wanka: Approximationsalgorithmen Eine Einführung, Teubner Verlag 2006.
- Hochbaum: Approximation Algorithms for NP-Hard Problems, Thomson Publishing, 1996.
- Ausiello, Crescenzi, Gambosi, Kann, Marchetti-Spaccamela, Protasi: Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties, Springer Verlag, 1999.
- Garey, Johnson: Computers and Intractibility, Freeman and Company, 1979.

Vazirani: Approximation Algorithms, Springer Verlag, 2001.

- Jansen, Margraf: Approximative Algorithmen und Nichtapproximierbarkeit, de Gruyter, 2008.
- Wanka: Approximationsalgorithmen Eine Einführung, Teubner Verlag 2006.
- Hochbaum: Approximation Algorithms for NP-Hard Problems, Thomson Publishing, 1996.
- Ausiello, Crescenzi, Gambosi, Kann, Marchetti-Spaccamela, Protasi: Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties, Springer Verlag, 1999.
- Garey, Johnson: Computers and Intractibility, Freeman and Company, 1979.

