Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Lehrstuhl für
INFORMATIK I
Algorithmen & Komplexität

Institut für Informatik

Algorithmische Graphentheorie

Sommersemester 2022

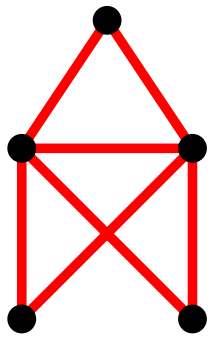
3. Vorlesung, Teil A

Rundreiseprobleme – Teil III

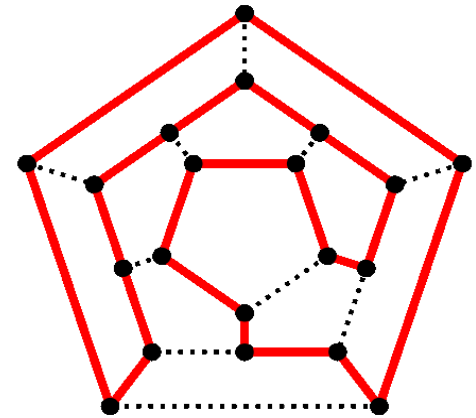
Prof. Dr. Alexander Wolff *Lehrstuhl für Informatik I*

Übersicht

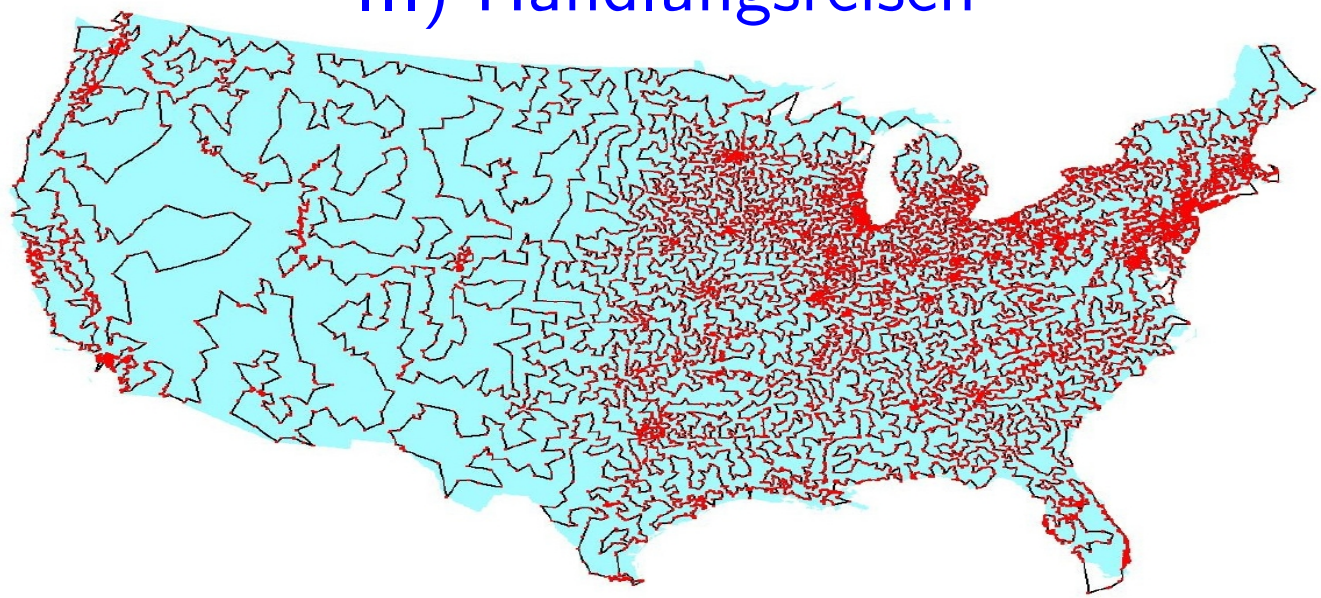
I) Eulerkreise



II) Hamiltonkreise



III) Handlungsreisen



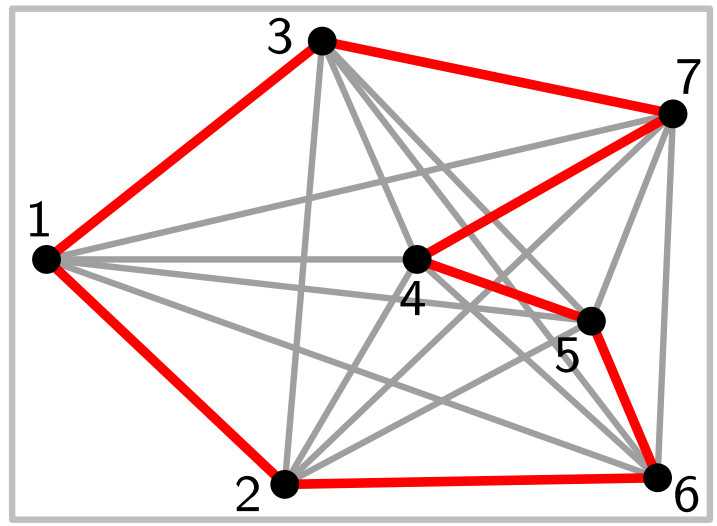
III) Handlungsreisen

Problem: *Traveling Salesman/Salesperson Problem (TSP)*

Gegeben: unger. vollständiger Graph $G = (V, E)$ mit Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

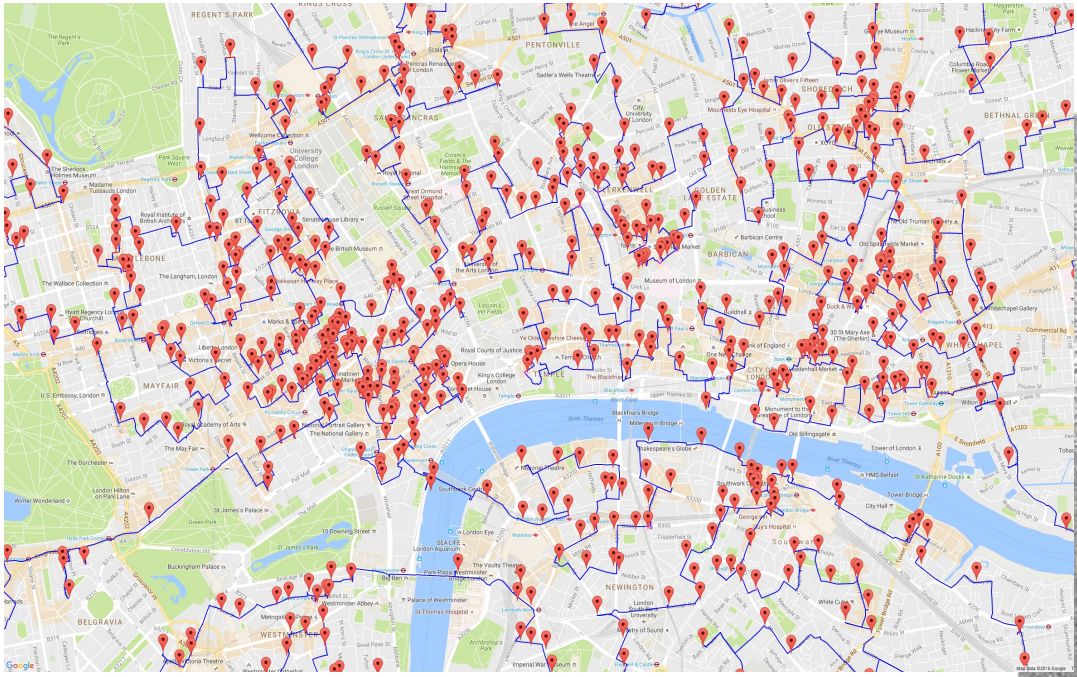
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K) := \sum_{e \in K} c(e)$.

Beispiel. $c \equiv d_{\text{Eukl.}}$



$$\begin{pmatrix} 0 & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} & c_{17} \\ c_{21} & 0 & c_{23} & c_{24} & c_{25} & c_{26} & c_{27} \\ c_{31} & c_{32} & 0 & c_{34} & c_{35} & c_{36} & c_{37} \\ c_{41} & c_{42} & c_{43} & 0 & c_{45} & c_{46} & c_{47} \\ c_{51} & c_{52} & c_{53} & c_{54} & 0 & c_{56} & c_{57} \\ c_{61} & c_{62} & c_{63} & c_{64} & c_{65} & 0 & c_{67} \\ c_{71} & c_{72} & c_{73} & c_{74} & c_{75} & c_{76} & 0 \end{pmatrix}$$

Beispielinstanzen



Alle 24.727 Pubs in Großbritannien.
(45.495,239 km)

Mona Lisa TSP Challenge.

Exakte Berechnung: Brute Force

Algorithmus:

- Für jede Permutation σ von $\langle 1, 2, \ldots, n \rangle$:

  Berechne die Kosten der Tour durch die Knoten $v_1, \ldots, v_n$ in dieser Reihenfolge:

  $c(\sigma) = c(v_{\sigma(n)} v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)} v_{\sigma(i+1)})$

- Gib die kürzeste Tour zurück.

Laufzeit: Anzahl Permutationen von n Objekten: $n!$

Hält man den 1. Knoten fest, so bleiben „nur“ $(n-1)!$ Permutationen.

Berechnung einer Tourlänge $c(\sigma)$: $O(n)$ Zeit.

Berechnung der nächsten Permutation: ???

Ang. ??? $= O(n)$, dann ist die Laufzeit $O(n!)$.

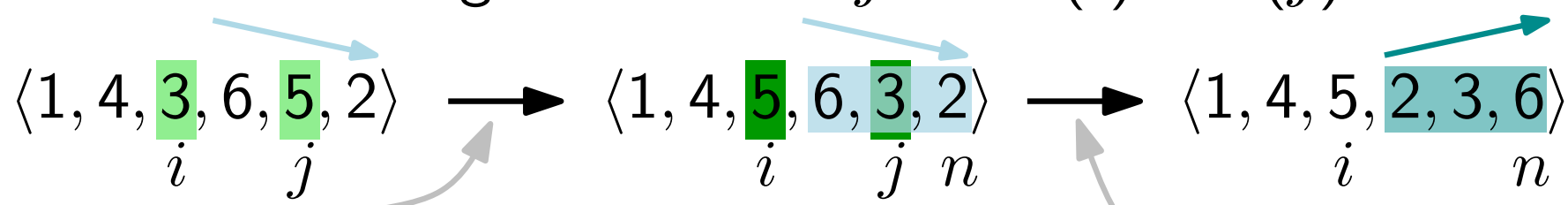
Speicher: $O(n)$ für aktuelle und bisher beste Permutation.

Wie iteriert man durch alle Permutationen?

Z.B. in lexikografischer Ordnung:
$\langle 1,2,3,4,5,6\rangle, \langle 1,2,3,4,6,5\rangle, \langle 1,2,3,5,4,6\rangle, \ldots, \langle 6,5,4,3,2,1\rangle$.

Für gegebene Permutation σ finde Nachfolger in $O(n)$ Zeit:

- Bestimme größten Index $i \in \{1,\ldots,n-1\}$ mit $\sigma(i) < \sigma(i+1)$.
- Falls nicht existiert, fertig ($\sigma =$ letzte Permutation).
- Sonst bestimme größten Index j mit $\sigma(i) < \sigma(j)$.



- Vertausche $\sigma(i)$ und $\sigma(j)$.
- Kehre die Teilfolge $\langle \sigma(i+1), \sigma(i+2), \ldots, \sigma(n)\rangle$ um.

Wie groß ist $n!$?

$$\underbrace{n/2 \cdot n/2 \cdot \ldots \cdot n/2}_{n/2 \text{ mal}} \leq n! = 1 \cdot 2 \cdot \ldots \cdot n \leq n \cdot n \cdot \ldots \cdot n = n^n$$

$$\Rightarrow \qquad 2^{n/2 \log_2 n/2} \leq n! \leq n^n = \left(2^{\log_2 n}\right)^n = 2^{n \log_2 n}$$

$$\Rightarrow \qquad \boxed{n! \in 2^{\Theta(n \log n)}}$$

Genauer: Sterlingformel [James Sterling, 1692–1770]

Für $n \to \infty$ gilt

$$n! \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n .$$

Noch genauer:

$$\sqrt{2\pi}\ \sqrt{n} \left(\frac{n}{e}\right)^n \leq n! \leq e\ \sqrt{n} \left(\frac{n}{e}\right)^n$$

Exakt, aber schneller: Dynamisches Programm

Wir beginnen alle Rundtouren im Knoten v_1.

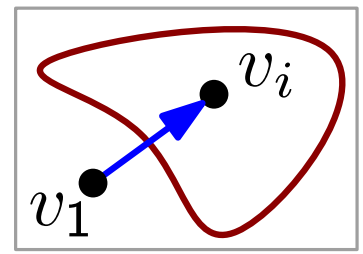
Für eine Knotenmenge $W \subseteq V \setminus \{v_1\}$ mit $v_i \in W$ definieren wir

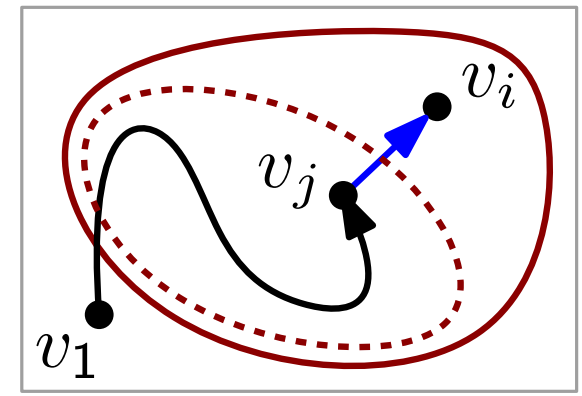
$\mathrm{OPT}[W, v_i]$:= optimale (kürzeste) Länge eines v_1-v_i-Wegs *durch alle Knoten in W*.

Idee für DP: *Definiere Wert einer opt. Lösung rekursiv!*

Dann gilt für $W = \{v_i\}$:

$$\mathrm{OPT}[W, v_i] = c(v_1, v_i)$$





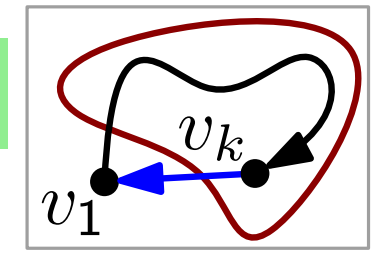
Und für W mit $\{v_i\} \subsetneq W$:

Letzter Knoten vor v_i

$$\mathrm{OPT}[W, v_i] = \min_{v_j \in W \setminus \{v_i\}} \left(\mathrm{OPT}[W \setminus \{v_i\}, v_j] + c(v_j, v_i)\right)$$

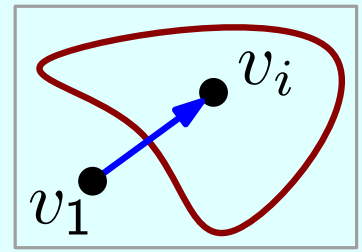
$$\Rightarrow \mathrm{OPT} = \min_{k \neq 1} \left(\mathrm{OPT}[V \setminus \{v_1\}, v_k] + c(v_k, v_1)\right)$$

Index des letzten Knotens vor v_1

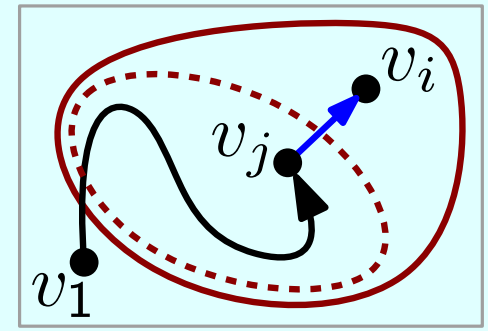


Der Algorithmus von Bellman & Held-Karp

BellmanHeldKarp(Knotenmenge V, Abstände $c\colon V \times V \to \mathbb{R}_{\geq 0}$)

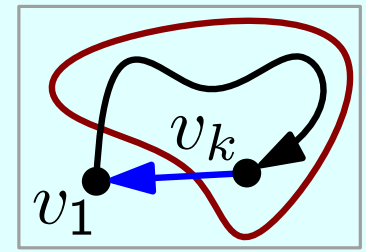


for $i = 2$ **to** n **do**
  $\mathrm{OPT}[\{v_i\}, v_i] = c(v_1, v_i)$



for $j = 2$ **to** $n-1$ **do**
  foreach $W \subseteq \{v_2, \ldots, v_n\}$ mit $|W| = j$ **do**
    foreach $v_i \in W$ **do**
      $\mathrm{OPT}[W, v_i] = \min_{v_j \in W \setminus \{v_i\}} \big(\mathrm{OPT}[W \setminus \{v_i\}, v_j] + c(v_j, v_i)\big)$

return $\min_{k \neq 1} \big(\mathrm{OPT}[V \setminus \{v_1\}, v_k] + c(v_k, v_1)\big)$



Laufzeit: Berechnung von $\mathrm{OPT}[W, v_i]$: $O(n)$

Wie viele Paare (W, v_i) mit $v_i \in W$ gibt's? $\leq n \cdot 2^{n-1}$

$\Rightarrow$ Gesamtlaufzeit $\in O\big(n^2 \cdot 2^n\big)$ **Speicher:** $O\big(n \cdot 2^n\big)$

Vergleich

	Brute Force	Bellman/Held-Karp
Laufzeit	$2^{\Theta(n \log n)}$	$O\left(n^2 \cdot 2^n\right)$
Speicher	$O(n)$	$O\left(n \cdot 2^n\right)$

Der Algorithmus von Held und Karp verringert also die Laufzeit zu Kosten des Speicherplatzverbrauchs.

Das bezeichnet man als Laufzeit-Speicherplatz-*Trade-Off*.

Bem. Nederlof hat vor kurzem [STOC 2020] einen randomisierten Algorithmus für *bipartites* Matching vorgeschlagen, der in $O(\text{poly}(n) \cdot 1.9999^n)$ Zeit läuft (falls man Matrizen schnell genug multiplizieren kann).

Bad News

Satz. TSP ist NP-schwer.

Beweis. Durch Reduktion vom Problem Hamiltonkreis.

Zu zeigen: wenn wir TSP effizient lösen könnten, dann auch HK.

in Polynomialzeit!

Gegeben: ungerichteter (i.A. *nicht* vollständiger) Graph $H = (V, E)$.

Def. vollständigen Graphen G mit Kosten c, so dass:

G hat billige TSP-Tour $\Leftrightarrow$ H hamiltonsch.

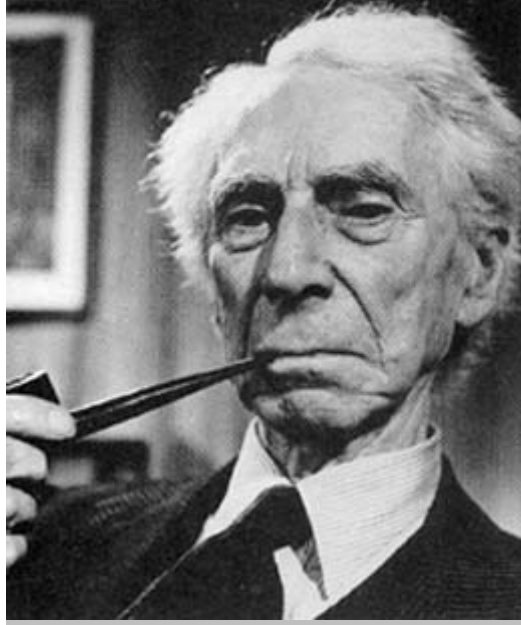
Nimm $G = \left(V, \binom{V}{2}\right)$ und für $u, v \in V$ setze

$$c(uv) = \begin{cases} 1, & \text{falls } uv \in E, \\ |V| & \text{sonst.} \end{cases}$$

Also: Optimale TSP-Tour kostet $|V| \Leftrightarrow H$ ham.

Was tun?

Bertrand Russell
(1872–1970)



„All exact science is dominated by the idea of approximation.“

Maximierungsproblem

Sei Π ein ~~*Minimierungsproblem*~~.

Sei ziel die *Zielfunktion* von Π: Lösung $\mapsto \mathbb{Q}_{\geq 0}$.

Sei γ eine Zahl $\not\geq 1$.

Güte von $\mathcal{A}$

Ein Algorithmus $\mathcal{A}$ heißt γ-*Approximation*, wenn

- $\mathcal{A}$ für jede Instanz I von Π eine Lösung $\mathcal{A}(I)$ berechnet, so dass

$$\frac{\text{ziel}(\mathcal{A}(I))}{\text{OPT}(I)} \not\leq \gamma$$

ziel(optimale Lösung)

Größe der Instanz I

- die Laufzeit von $\mathcal{A}$ polynomiell in $|I|$ ist.

z.B. TSP

ziel $\equiv c$

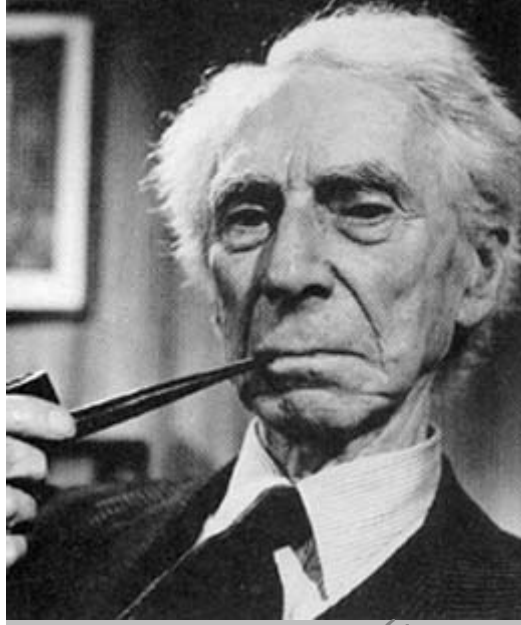
γ-Approx. für TSP liefert Tour die höchstens γ mal so teuer ist wie billigste Tour.

poly($|V|$)

Was tun?

„All exact science is dominated by the idea of approximation.“

Bertrand Russell
(1872–1970)



Maximierungsproblem

Sei Π ein ~~*Minimierungsproblem*~~.

Sei ziel die *Zielfunktion* von Π: Lösung $\mapsto \mathbb{Q}_{\geq 0}$.

Sei γ eine Zahl $\not\geq 1$.

Güte von $\mathcal{A}$

Ein Algorithmus $\mathcal{A}$ heißt γ-*Approximation*, wenn

- $\mathcal{A}$ für jede Instanz I von Π eine Lösung $\mathcal{A}(I)$ berechnet, so dass

$$\frac{\text{ziel}(\mathcal{A}(I))}{\text{OPT}(I)} \not\leq \gamma$$

ziel(optimale Lösung)

- die Laufzeit von $\mathcal{A}$ polynomiell in $|I|$ ist.

z.B. LSP = { longest simple path

ziel $\equiv$ Länge

γ-Approx. für LSP liefert einfachen Pfad, der *mindestens* γ mal so lang ist wie der längste.

poly($|V|$)

Bad News II

D.h. für kein $\gamma \geq 1$ gibt's eine γ-Approximation für TSP (außer P=NP).

Satz. TSP ist NP-schwer zu approximieren!

Beweis. Durch Reduktion vom Problem Hamiltonkreis.

Zu zeigen: wenn wir TSP ~~effizient lösen~~ mit Güte γ approximieren könnten, dann könnten wir auch HK eff. lösen.

Gegeben: ungerichteter (i.A. *nicht* vollständiger) Graph $H = (V, E)$.

Def. vollständigen Graphen G mit Kosten c, so dass:

G hat billige TSP-Tour $\Leftrightarrow$ H hamiltonsch.

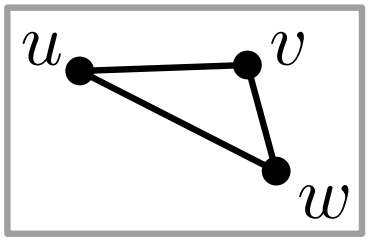
Nimm $G = \left(V, \binom{V}{2}\right)$ und für $u, v \in V$ setze

$$c(uv) := \begin{cases} 1, & \text{falls } uv \in E, \\ |V| \cdot \gamma & \text{sonst.} \end{cases}$$

$\Rightarrow$ ~~optimale~~ TSP-Tour (mit Güte γ) kostet $\leq \gamma \cdot |V| \Leftrightarrow H$ ham.

Was tun? – Mach das Problem leichter!

Problem: *Metrisches Traveling Salesman Problem* *(Δ-TSP)*

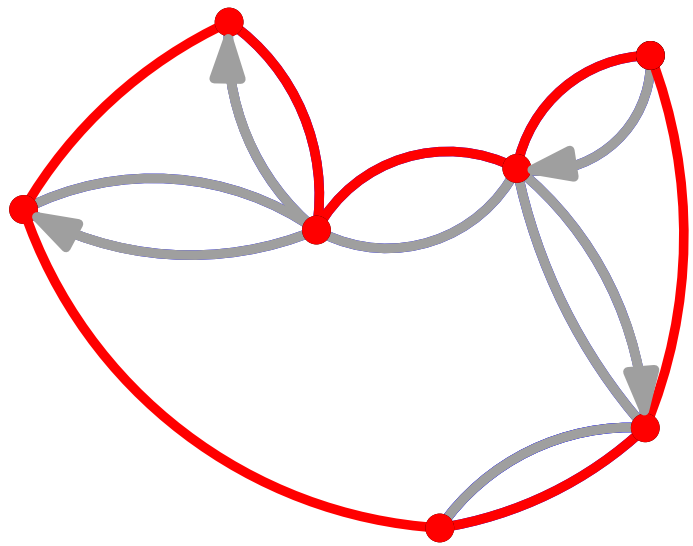


Gegeben: unger. vollständiger Graph $G = (V, E)$
mit Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$,
die die Dreiecksungleichung erfüllen,
d.h. $\forall u, v, w \in V : c(u, w) \leq c(u, v) + c(v, w)$.

Gesucht: Hamiltonkreis in G mit minimalen Kosten.

Satz. Es gibt eine 2-Approximation für Δ-TSP.

Beweis.



Algorithmus:

Berechne min. Spannbaum MSB.

Verdopple MSB ⇒ ergibt Kreis!

Durchlaufe den Kreis.

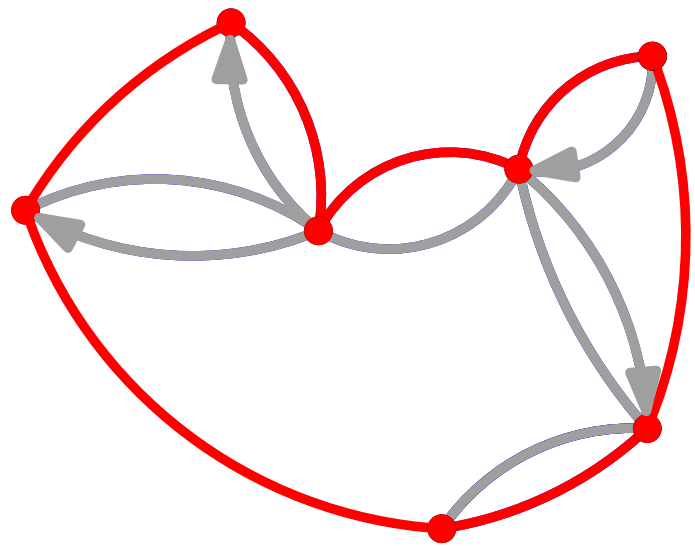
Überspringe besuchte Knoten.

Füge „Abkürzungen“ ein.

Analyse

Satz. Es gibt eine 2-Approximation für Δ-TSP.

Beweis.



1. *Algorithmus*

Berechne MSB von G.

Verdopple MSB $\Rightarrow$ ergibt Kreis!

Durchlaufe den Kreis.

Überspringe besuchte Knoten.

Füge „Abkürzungen“ ein.

2. *Analyse*

$$c(\text{ALG}) \leq c(\text{Kreis}) = 2 \cdot c(\text{MSB}) \leq 2 \cdot \text{OPT}$$

Dreiecksungleichung

Optimale TSP-Tour minus eine Kante ist (i.A. nicht minimaler) Spannbaum!!

Die „Kunst“ der unteren Schranke: $c(\text{min. Spannbaum}) \leq c(\text{TSP-Tour})$