Julius-Maximilians-UNIVERSITÄT WÜRZBURG

Lehrstuhl für INFORMATIK I Algorithmen & Komplexität

Institut für Informatik

Algorithmen und Datenstrukturen

Wintersemester 2022/23

24. Vorlesung

Der Handlungsreisende

Prof. Dr. Alexander Wolff

Lehrstuhl für Informatik I

Das Problem

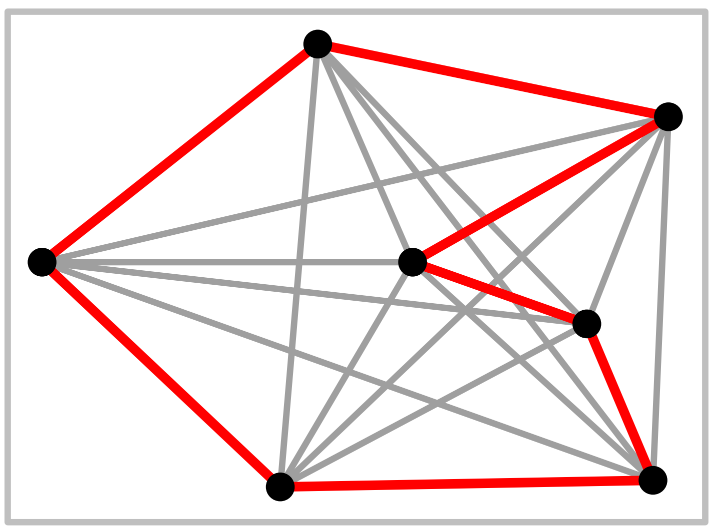
Definition. *Traveling Salesperson Problem (TSP)*

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 $c(K) := \sum_{e \in K} c(e)$.

(Ein HK besucht jeden Knoten genau $1\times$.)

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



Problem.

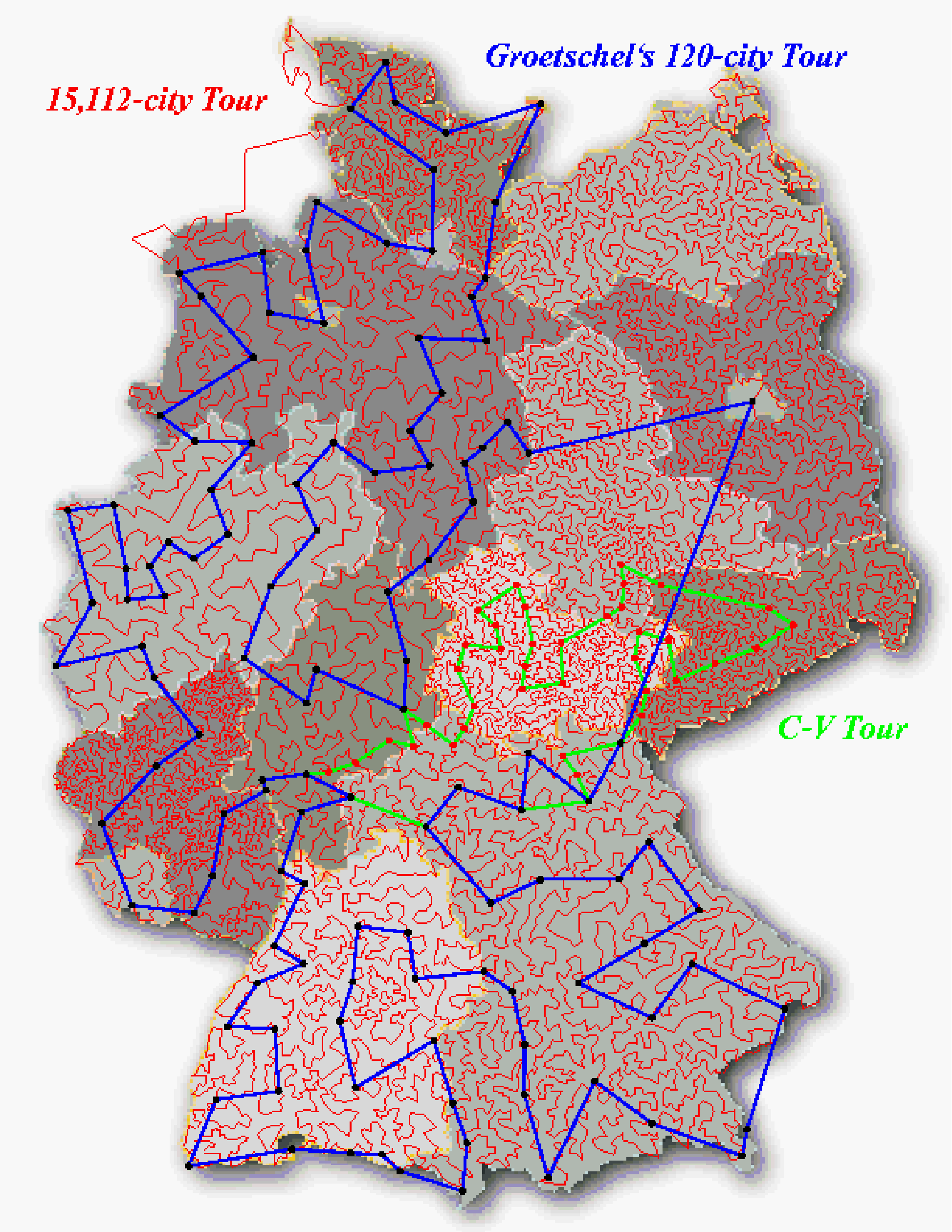
- TSP ist NP-schwer
- und schwer zu approximieren. 

Etwas Geschichte

Der Handlungsreisende – wie er sein soll und was er zu thun hat, um Aufträge zu erhalten und eines glücklichen Erfolgs in seinen Geschäften gewiss zu sein.
Von einem alten Commis-Voyageur [1832]

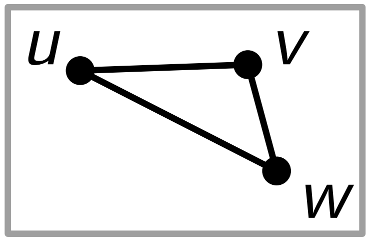
Rekord I:
optimale 120-Städte-Tour [Groetschel, 1977]

Rekord II:
optimale 15.112-Städte-Tour [Applegate, Bixby, Chvátal, Cook 2001]



Was tun? – Mach das Problem leichter!

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

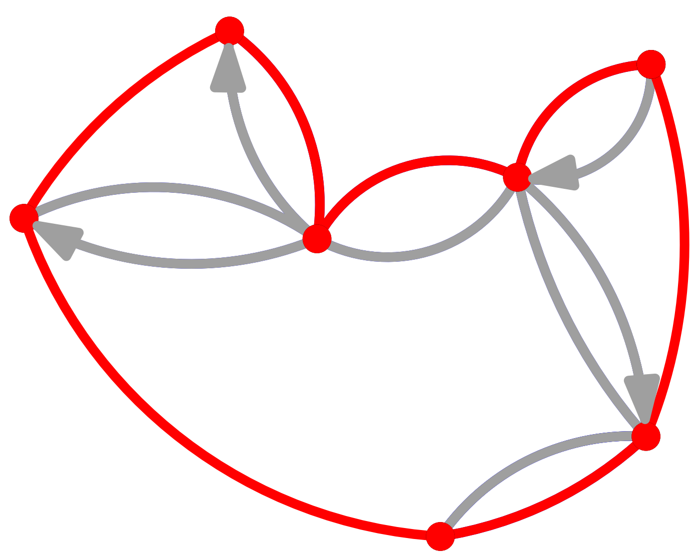


Gegeben: unger. vollständiger Graph $G = (V, E)$
mit Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$,
die die Dreiecksungleichung erfüllen,
d.h. $\forall u, v, w \in V\colon c(u, w) \leq c(u, v) + c(v, w)$.

Gesucht: Hamiltonkreis in G mit minimalen Kosten.

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

Beweis.



Algorithmus:

Berechne min. Spannbaum MSB.

Verdopple MSB $\Rightarrow$ ergibt Kreis!

Durchlaufe den Kreis.

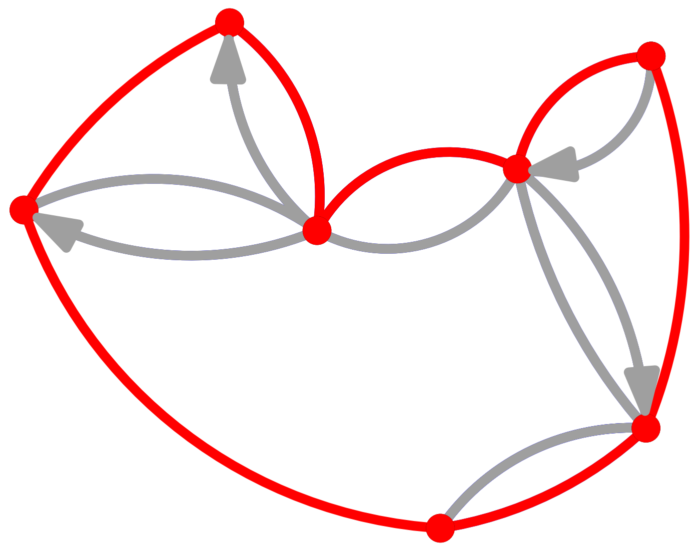
Überspringe besuchte Knoten.

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: $c(\text{min. Spannbaum}) \leq c(\text{TSP-Tour})$

Exakte Berechnung: Brute Force

Algorithmus:

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

  Berechne die Kosten der Tour durch die Knoten $v_1, \dots, v_n$ in dieser Reihenfolge:
  $c(\sigma) = \sum_{i=1}^{n-1} c(v_{\sigma(i)} v_{\sigma(i+1)}) + c(v_{\sigma(n)} v_{\sigma(1)})$

- 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: bisher beste, aktuelle & nächste Permutation.

Wie iteriert man durch alle Permutationen?

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

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

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

$$\langle 1,4,\underset{i}{3},6,\underset{j}{5},2\rangle \longrightarrow \langle 1,4,\underset{i}{5},6,\underset{j}{3},\underset{n}{2}\rangle \longrightarrow \langle 1,4,\underset{i}{5},2,3,\underset{n}{6}\rangle$$

- Vertausche $\sigma(i)$ und $\sigma(j)$.
- Kehre die Teilfolge $\langle \sigma(i+1), \sigma(i+2), \dots, \sigma(n)\rangle$ 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: Stirlingformel [James Stirling, 1692–1770]

Für $n \to \infty$ gilt

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

Noch genauer:

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

Exakter TSP-Algorithmus: Schneller per DP!

Wir beginnen alle Rundtouren im Knoten v_1.

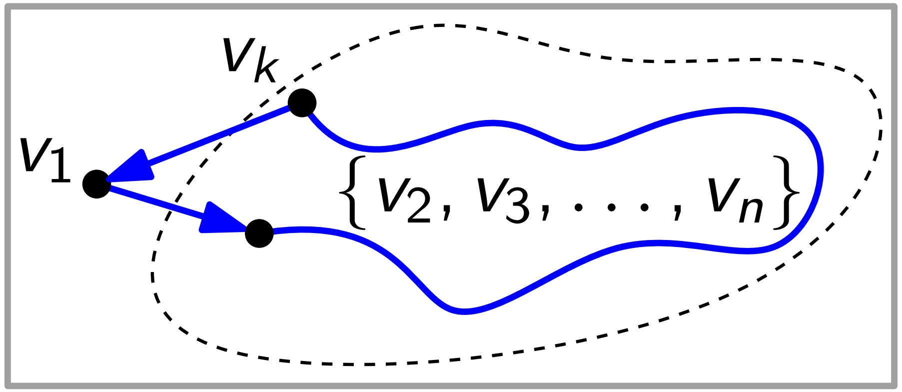
Für eine Knotenmenge $W \subseteq V \setminus \{v_1\}$ mit $v_i \in W$ definiere:

$T[W, v_i]$:= optimale (kürzeste) Länge eines v_1-v_i-Wegs *durch alle Knoten in* W.

Schritt 2 für DP: *Definiere Wert einer opt. Lösung rekursiv!*

Dann gilt für $W = \{v_i\}$, $i > 1$:

$$T[W, v_i] = c(v_1, v_i)$$



Und für W mit $|W| \geq 2$, $v_i \in W$:

Letzter Knoten vor v_i

$$T[W, v_i] = \min_{v_j \in W \setminus \{v_i\}} T[W \setminus \{v_i\}, v_j] + c(v_j, v_i)$$

$$\Rightarrow \quad \mathrm{OPT} = \min_{k \neq 1} T[\{v_2, v_3, \ldots, v_n\}, v_k] + c(v_k, v_1)$$

Index des letzten Knotens vor v_1

Der Algorithmus von Bellman, Held & Karp

Schritt 3 für DP: *Berechne Wert einer opt. Lsg. (hier: bot.-up)!*

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

for $i = 2$ **to** n **do**
  $T[\{v_i\}, v_i] = c(v_1, v_i)$
for $j = 2$ **to** $n-1$ **do**
  foreach $W \subseteq \{v_2, \dots, v_n\}$ mit $|W| = j$ **do**
    foreach $v_i \in W$ **do**
      $T[W, v_i] = \min_{v_j \in W \setminus \{v_i\}} T[W \setminus \{v_i\}, v_j] + c(v_j, v_i)$
return $\min_{k \neq 1} T[\{v_2, v_3, \dots, v_n\}, v_k] + c(v_k, v_1)$

Laufzeit: Berechnung von $T[W, v_i]$: $O(n)$ Zeit
Wie viele Paare (W, v_i) mit $v_i \in W$ gibt's? $\leq 2^{n-1} \cdot n$
$\Rightarrow$ Gesamtlaufzeit $\in O\left(n^2 \cdot 2^n\right)$ **Speicher:** $O\left(n \cdot 2^n\right)$

Vergleich

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

Der Bellman-Held-Karp-Algorithmus verringert also die Laufzeit zu Kosten des Speicherplatzverbrauchs.

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

*) Wie wäre es, wenn wir im DP nicht *ganz* $T[\cdot,\cdot]$ speichern?

Für $T[W,\cdot]$ brauchen wir nur alle $T[W',\cdot]$ mit $|W'| = |W| - 1$.

Welches j maximiert $\binom{n}{j}$? $j = \frac{n}{2}$.

Wie groß ist $\binom{n}{n/2}$? In $\Theta(2^n/\sqrt{n})$.

Richard M. Karp
(*1935)

Richard E. Bellman
(1920–1984)