Polynomiales Approximationsschema für Euklidisches TSP

Wir beschreiben das von Arora (1998) gefundene PTAS für Euklidische TSP. Die Darstellung folgt im Wesentlichen (Korte and Vygen, 2006, Kapitel 21).

Grundlagen: randomisierte Algorithmen

Zunächst einige einfache stochastische Grundlagen, um die wir nicht herum kommen.

Monte-Carlo-Algorithmus. Der randomisierte Algorithmus, den wir betrachten, benutzt zufällige Parameter, um in (deterministisch) polynomialer Zeit mit einer gewissen Mindestwahrscheinlichkeit erfolgreich eine „Lösung“ zu berechnen. „Lösung“ heißt dabei hier: eine zulässige Lösung (d. h. Tour) mit Gütegarantie $1 + \varepsilon$.

Derandomisierung. Gegeben sei ein Algorithmus mit „Erfolgswahrscheinlichkeit“ $p > 0$. Dann kann man die Erfolgswahrscheinlichkeit erhöhen, indem man ihn mehrfach hintereinander mit stochastisch unabhängiger Wahl der Parameter ausführt. Ist k die Anzahl der Ausführungen, dann bekommt man so einen (immer noch randomisierten) Algorithmus mit $(1-(1-p)^k)$-wahrscheinlicher Lösung. (in unserem Fall: Gütegarantie). Beispiel: $p = 1/2$, dann ist die Erfolgswahrscheinlichkeit bei 10-maligem Hintereinanderausführen bereits

$$1 - \frac{1}{2^{10}} \;>\; 0.999\%$$

Können die zufälligen Parameter nur endlich viele Werte annehmen (insbesondere ist dann die Zufallsverteilung der Parameterwahlen diskret), dann liefert das Ausführen des Algorithmus mit allen möglichen Parameterwahlen einen *derandomisierten* (deterministischen) Algorithmus. Allerdings vervielfacht sich dabei natürlich die Laufzeit (i. d. R. um einen inakzeptablen exponentiellen Faktor), d. h. der Algorithmus ist dann ggf. nicht mehr polynomial (in unserem Fall aber schon!).

Markov-Ungleichung. Ist X eine reellwertige, nichtnegative Zufallsvariable mit Erwartungswert E und $a \geq 1$, dann gilt

$$P(X \geq aE) \leq \frac{1}{a}.$$

ε-diskretisierte Euklidische TSPs

$d(v, w)$ bezeichne die Euklidische Distanz zwischen zwei Punkten $v, w \in \mathbb{R}^2$. Zunächst überlegt man sich, dass es reicht, ein PTAS für Euklidische TSPs mit speziellen, diskreten Eigenschaften anzugeben.

Definition (ε-diskretisiertes Euklidisches TSP). *Sei $\varepsilon > 0$ gegeben. Ein Euklidisches TSP, definiert durch eine Punktemenge $V \subset \mathbb{R}^2$, $|V| = n$ heißt* ε-diskretisiert*, falls gilt:*

(a) *Alle Punkte in V haben ganzzahlige und ungerade Koordinaten;*

(b) $\max_{v,w \in V} d(v, w) \leq \frac{64n}{\varepsilon} + 16$*;*

(c) $\min_{v,w \in V} d(v, w) \geq 8$. △

Offenbar gilt: V ε-diskretisiert $\implies$ V ε'-diskretisiert für jedes $\varepsilon' \leq \varepsilon$.

Satz. *Angenommen es gibt ein PTAS für ε-diskretisierte Euklidische TSPs (d. h. es gibt einen Algorithmus, der für jedes $\varepsilon > 0$ und jedes ε-diskretisierte Euklidische TSP in polynomialer Zeit (in $|V|$) eine Tour mit Güte $1+\varepsilon$ liefert). Dann gibt es auch ein PTAS für beliebige Euklidische TSPs.* △

Beweis. Sei $V \subset \mathbb{R}^2$, $|V| = n \geq 3$, sowie $\varepsilon > 0$ gegeben. Dann müssen wir eine Tour durch V mit Güte $1 + \varepsilon$ berechnen können.

Setze $L := \max_{v,w \in V} d(v, w)$ und definiere eine modifizierte Instanz durch

$$V' := \left\{ \left(1 + 8 \left\lfloor \frac{8n}{\varepsilon L} v_x \right\rfloor, 1 + 8 \left\lfloor \frac{8n}{\varepsilon L} v_y \right\rfloor \right) x \mid (v_x, v_y) \in V \right\}.$$

V' enthält höchstens so viele Punkte wie V (einige könnten beim Runden zusammenfallen) und es gilt (wilde Rechnerei):

$$\max_{v,w \in V'} d(v, w) \leq \frac{64n}{\varepsilon} + 16.$$

(a) und (c) gelten offensichtlich für V', d. h. V' ist ε-diskretisiert und damit auch $\varepsilon/2$-diskretisiert. Nach Annahme können wir jetzt eine Tour T' finden mit $c(T') \leq (1 + \varepsilon/2) \cdot c(T'_{\text{opt}})$, wobei T'_{opt} eine optimale Tour von V' ist. Aus T' kann man eine Tour T durch V konstruieren, mit Gesamtkosten (noch mehr Rechnerei)

$$c(T) \leq \left(\frac{c(T')}{8} + 2n \right) \frac{\varepsilon L}{8n}.$$

Außerdem gilt für eine optimale Tour T_{opt} durch V (ähnliche Rechnerei):

$$c(T'_{\text{opt}}) \leq 8 \left(\frac{8n}{\varepsilon L} c(T_{\text{opt}}) + 2n \right) \quad \text{und} \quad c(T_{\text{opt}}) \geq 2L.$$

Insgesamt bekommt man (für $\varepsilon \leq 4$):

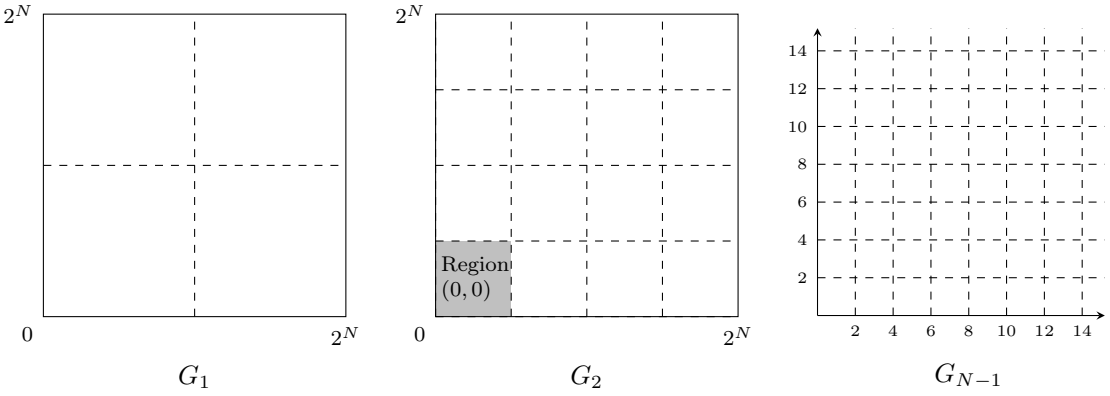
$$\begin{aligned} c(T) &\leq \frac{\varepsilon L}{8n}\left(\left(1+\frac{\varepsilon}{2}\right)\left(\frac{8n}{\varepsilon L}c(T_{\text{opt}})+2n\right)+2n\right) \\ &= \left(1+\frac{\varepsilon}{2}\right)c(T_{\text{opt}})+\frac{\varepsilon L}{2}+\frac{\varepsilon^2 L}{8} \\ &\leq \left(1+\frac{\varepsilon}{2}\right)c(T_{\text{opt}})+\frac{\varepsilon}{4}\,c(T_{\text{opt}})+\frac{\varepsilon^2}{16}\,c(T_{\text{opt}}) \;\leq\; (1+\varepsilon)\,c(T_{\text{opt}}) \end{aligned}$$ □

Notationen und Hilfssätze

Der Satz im letzten Abschnitt stellt sicher, dass wir nur noch ε-diskretisierte Euklidische TSPs betrachten müssen, für gegebenes $\varepsilon > 0$. Außerdem kann man natürlich eine gegebene Instanz beliebig als ganzes verschieben ohne dass sich die Längen von Touren ändern, d. h. wir können ohne Einschränkung annehmen, dass alle Punkte unseres gegebenen ε-diskretisierten Euklidischen TSP im Quadrat $[0, 2^N] \times [0, 2^N]$ liegen, wobei $N := \lceil \log L \rceil + 1$ mit $L = \max_{v,w \in V} d(v, w)$ ist. Beachte: Unter diesen Voraussetzungen ist $N \in O(\log \frac{n}{\varepsilon})$, wegen (b).

Der Algorithmus berechnet (mittels dynamischer Programmierung) Touren in Teilquadraten des Ausgangsquadrats und setzt diese dann zusammen. Dazu definieren wir verschieden feine Unterteilungen des Ausgangsquadrats. Im Folgenden bezeichne $i = 1, \ldots, N-1$ jeweils die *Ebene* der Unterteilung. Setze

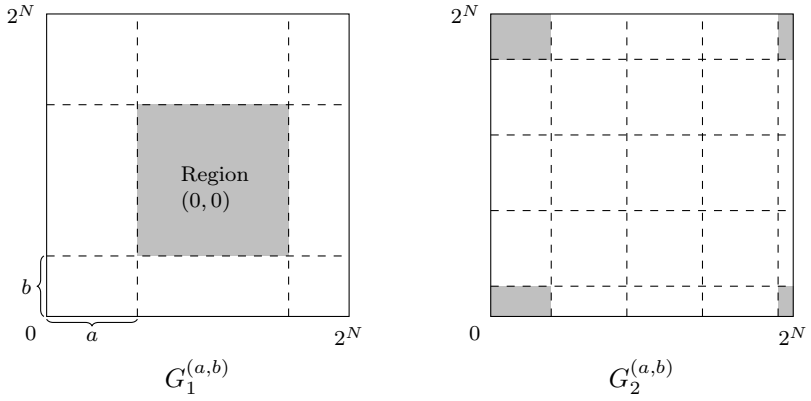
$$\begin{aligned} X_i &:= \left\{\{(x, k \cdot 2^{N-i}) \mid 0 \leq x \leq 2^N\} \mid k = 0, \ldots, 2^i - 1\right\}, \\ Y_i &:= \left\{\{(j \cdot 2^{N-i}, y) \mid 0 \leq y \leq 2^N\} \mid j = 0, \ldots, 2^i - 1\right\}, \\ G_i &:= X_i \cup Y_i. \end{aligned}$$



Die Elemente von $G_i \setminus G_{i-1}$ (mit $G_0 := \emptyset$) sind die *Linien auf Ebene* i und sie teilen das Gesamtquadrat in $2^i \cdot 2^i = 2^{2i}$ Regionen auf.

In Wirklichkeit betrachten wir das eben definierte Gitter, allerdings um einen (später zufälligen) Vektor verschoben: Sind $a, b \in \{0, 2, \dots 2^N - 2\}$, dann setze

$$\begin{aligned} X_i^{(b)} &:= \left\{ \{(x, y + b \bmod 2^N) \mid (x, y) \in \ell\} \mid \ell \in X_i \right\}, \\ Y_i^{(a)} &:= \left\{ \{(x + a \bmod 2^N, y) \mid (x, y) \in \ell\} \mid \ell \in Y_i \right\}, \\ G_i^{(a,b)} &:= X_i^{(b)} \cup Y_i^{(a)}. \end{aligned}$$



Die 2^{2i} Regionen der Ebene i sind dann

$$R_{i,(j,k)}^{(a,b)} = \Big\{ (x, y) \in [0, 2^N) \times [0, 2^N) \mid (x - a - j \cdot 2^{N-i}) \bmod 2^N < 2^{N-i}, \\ (y - b - k \cdot 2^{N-i}) \bmod 2^N < 2^{N-i} \Big\},$$

und diese partitionieren, zusammen mit den Linien in $G_i^{(a,b)}$ das Anfangsquadrat. Beobachtungen:

- Das (letzte) Gitter auf Ebene $N - 1$ hängt nicht von a und b ab: $G_{N-1}^{(a,b)} = G_{N-1}$ für alle a, b.
- Auf Ebenen $i < N - 1$ können Regionen aus zwei oder vier disjunkten Teilen bestehen.
- Kein Punkt des ε-diskretisierten Euklidischen TSP liegt auf irgendeiner Linie (weil die Linien immer gerade y- bzw. x-Koordinaten haben).
- Jede Region auf Ebene $N - 1$ ist ein Quadrat mit Seitenlänge 2 und enthält höchstens einen Punkt des ε-diskretisierten Euklidischen TSP.

Im Algorithmus wird rekursiv ein Baum der Regionen aufgebaut, die Punkte aus V enthalten, und für jeden Knoten dieses Baums muss ein Teilproblem gelöst werden. Zunächst kommen aber noch zwei Lemmas, die im weiteren nützlich sind.

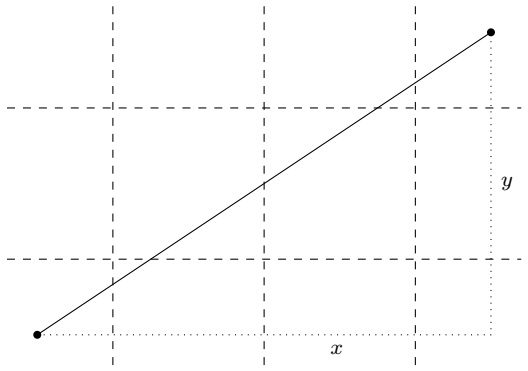
Für einen Polygonzug T (z. B. eine Tour) und ein Streckensegment ℓ (z. B. eine Linie in G_i) bezeichnen wir mit $\mathrm{cr}(T, \ell)$ die Anzahl der Überkreuzungen von T und ℓ. Beachte: $\mathrm{cr}(T, \ell)$ kann i. A. größer sein als die Anzahl der Schnittpunkte.

Lemma (Kreuzungslemma). *Sei T_{opt} eine optimale Tour eines ε-diskretisierten Euklidischen TSP, gegeben durch $V \subset [0, 2^N]^2$. Dann gilt*

$$\sum_{\ell \in G_{N-1}} \text{cr}(T_{\text{opt}}, \ell) \leq c(T_{\text{opt}}).$$

△

Beweis. Betrachte eine Kante e von T_{opt} mit Länge c_e, sowie horizontalem (bzw. vertikalem) Abstand der Endpunkte x (bzw. y). Linien der Ebene $N-1$ haben Abstand 2 voneinander und werden daher von e maximal $\frac{x}{2} + 1 + \frac{y}{2} + 1$ oft geschnitten.



Wegen $x + y \leq \sqrt{2} c_e$ (Pythagoras und binomische Formel) und $c_e \geq 8$ (die Instanz ist ε-diskretisiert) folgt

$$\sum_{\ell \in G_{N-1}} \text{cr}(e, \ell) \;\leq\; \frac{x+y}{2} + 2 \;\leq\; \frac{\sqrt{2}}{2} c_e + 2 \;\leq\; c_e.$$

Summieren über alle Kanten der Tour T_{opt} ergibt:

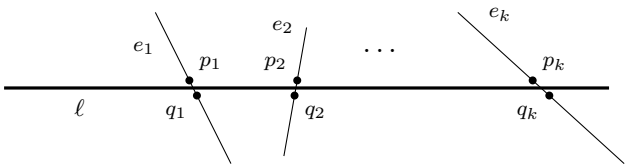
$$\sum_{\ell \in G_{N-1}} \text{cr}(T_{\text{opt}}, \ell) \leq \sum_{e \in T_{\text{opt}}} c_e = c(T_{\text{opt}}).$$

□

Wir brauchen noch ein Lemma, das es erlaubt, bestimmte Touren nach bestimmten Regeln zu modifizieren.

Lemma (Patching-Lemma). *Sei T eine Tour für ein Euklidisches TSP, gegeben durch $V \subset \mathbb{R}^2$, und ℓ ein Streckensegment der Länge s, das keinen Punkt von V enthält. Dann gibt es eine Tour T' für V mit $c(T') \leq c(T) + 6s$, die ℓ höchstens zweimal kreuzt.* △

Beweis. Angenommen, T schneidet ℓ mit $k \geq 3$ Kanten $e_1, \ldots, e_k$. Wir unterteilen jede dieser Kanten durch zwei neue Punkte p_i, q_i, $1 \leq i \leq k$, die auf verschiedenen Seiten, aber sehr nahe an ℓ liegen.



Setze $t := \lfloor (k-1)/2 \rfloor$, d. h. $2t = k-1$ falls k ungerade ist und $2t = k-2$ falls k gerade ist. Nun konstruieren wir eine neue Tour T' folgendermaßen:

1. Entferne alle Teilstücke zwischen p_i und q_i für $1 \leq i \leq 2t$.
2. Füge eine neue kürzeste Tour innerhalb der Punkte $p_1, \ldots, p_k$ und ein minimales perfektes Matching innerhalb der Punkte $p_1, \ldots, p_{2t}$ hinzu.
3. Füge eine neue kürzeste Tour innerhalb der Punkte $q_1, \ldots, q_k$ und ein minimales perfektes Matching innerhalb der Punkte $q_1, \ldots, q_{2t}$ hinzu.
4. In dem so entstehenden Eulerschen Graphen (aufgrund der hinzugefügten Matchings haben alle Knoten geraden Grad!) kann man jetzt eine Tour durch die Punktmenge $V \cup \{p_1, \ldots, p_k\} \cup \{q_1, \ldots, q_k\}$ finden.
5. Durch Abkürzen in der im letzten Schritt gefundenen Tour in den p_i und q_i ergibt sich die neue Tour T' für V.

Die Kantengewichte der zwischenzeitlich neu hinzugefügten Strecken werden natürlich jeweils gemäß Euklidischen Distanzen gesetzt. Aufgrund der Wahl der Punkte sind die Gesamtlängen der hinzugefügten Touren durch die p_i bzw. q_i höchstens $2s$ und die Gesamtlängen der Matchings höchstens s. Insgesamt sind also Kanten vom Gesamtgewicht höchstens $6s$ hinzugefügt worden, und das Gesamtgewicht wird in den letzten beiden Schritten allenfalls verringert. Die so konstruierte Tour T' schneidet dann ℓ höchstens noch in den Kanten e_{k-1} und e_k (falls k gerade war) bzw. e_k (falls k ungerade war). □

Aroras Algorithmus funktionert so, dass eine Tour durch V, plus weitere Punkte gefunden wird, die (wahrscheinlich) die gewünschte Güteschranke einhält. Die zusätzlichen Punkte sind so gewählt, dass daraus eine (die tatsächlich gesuchte) Tour durch V konstruiert werden kann. Dazu führen wir zunächst noch ein paar Abkürzungen und Bezeichnungen ein, bevor wir den zentralen Satz, der alles zum laufen bringt, und schließlich den Algorithmus selbst angeben.

Wir setzen

$$C := 7 + \left\lceil \frac{36}{\varepsilon} \right\rceil \quad \text{und} \quad P := N \left\lceil \frac{6}{\varepsilon} \right\rceil .$$

Für eine horizontale Linie $\ell \in G_i^{(a,b)}$ auf Ebene i (die „k-te von b aus gesehen“) nennen wir die Punkte

$$\left((a + \frac{h}{P} 2^{N-i}) \bmod 2^N, (b + k \cdot 2^{N-i}) \bmod 2^N \right) , \qquad h = 0, \ldots, P \cdot 2^i$$

die *Portale auf* ℓ. Analog definiere Portale auf vertikalen Linien in $G_i^{(a,b)}$. Auf einer Linie auf Ebene i liegen also insgesamt $P \cdot 2^i + 1$ Portale gleichmäßig verteilt, so dass zwei benachbarte Portale Abstand $2^{N-i}/P$ voneinander haben.

Eine *Steiner-Tour* ist ein Streckenzug (der sich auch überkreuzen kann) in $[0, 2^N]^2$, der alle Punkte von V enthält, und der sich mit den Linien in $G_i^{(a,b)}$ für alle i nur in Portalen schneidet. Eine Steiner-Tour heiß *leicht*, wenn sie für jedes i und jede von $G_i^{(a,b)}$ definierte Region R jede begrenzende Kante von R höchstens C mal schneidet.

Satz (Arora 1998). *Sei ein ε-diskretisiertes Euklidisches TSP durch $V \subset [0, 2^N]^2$ definiert und T_{opt} eine optimale Tour. Sind $a, b \in \{0, 2, \ldots, 2^N - 2\}$ zufällig gewählt, dann existiert mit Wahrscheinlichkeit mindestens $1/2$ eine leichte Steiner-Tour T mit Gesamtlänge $c(T) \leq (1 + \varepsilon)c(T_{\text{opt}})$.* △

Beweis. Wir konstruieren eine Steiner-Tour aus T_{opt} auf folgende Weise. Zunächst wird an jedem Kreuzungspunkt von T_{opt} mit irgendeiner der Linien aus G_{N-1} ein zusätzlicher Punkt definiert. Anschließend wird jeder dieser zusätzlichen Punkte in das jeweils nächstliegende Portal verschoben. Daraus ergibt sich eine Steiner-Tour T', für die man die erwartete Verlängerung gegenüber T_{opt} folgendermaßen abschätzen kann: Sei ℓ irgendeine Linie aus G_{N-1} zufällig (gleichverteilt) gewählt. Dann ist die Wahrscheinlichkeit, dass ℓ auf Ebene i ist, gleich

$$p(\ell, i) := \begin{cases} 2^{i-N} & \text{falls } i > 1 \\ 2^{2-N} & \text{falls } i = 1 \end{cases}$$

und jeder Zusatzpunkt muss um höchstens $2^{N-i}/2P$ verschoben werden, um in einem Portal zu landen. Damit ist die erwartete Verlängerung höchstens

$$\begin{aligned} \sum_{\ell \in G_{N-1}} \sum_{i=1}^{N-1} p(\ell, i) \operatorname{cr}(T_{\text{opt}}, \ell) \cdot 2 \cdot \frac{2^{N-i}}{2P} &= \sum_{\ell \in G_{N-1}} \left(\frac{2}{P} \operatorname{cr}(T_{\text{opt}}, \ell) + \sum_{i=2}^{N-1} \frac{1}{P} \operatorname{cr}(T_{\text{opt}}, \ell) \right) \\ &= \sum_{\ell \in G_{N-1}} \frac{N}{P} \operatorname{cr}(T_{\text{opt}}, \ell). \end{aligned}$$

Als nächstes benutzen wir das Patching-Lemma, um die Steiner-Tour T' zu einer leichten Steiner-Tour zu machen. Für alle $i = N-1, \ldots, 1$ (in dieser Reihenfolge) wenden wir das Lemma zunächst auf alle horizontalen Linienstücke zwischen den Schnittpunkten auf Ebene i an, die von T' mehr als $C - 4$ mal geschnitten werden; danach auf alle vertikalen Linien in der gleichen Weise. Nach jeder Anwendung des Patching-Lemmas wird das entsprechende Linienstück nur noch zwei- oder viermal (falls es zu einer unzusammenhängenden Region gehört) geschnitten. Während der gesamten Prozedur könnten allerdings neue Schnittpunkte hinzugekommen sein, um die wir uns aber später kümmern.

Die Anzahl der Anwendungen des Patching-Lemmas auf eine Linie ℓ ist höchstens

$$\frac{\operatorname{cr}(T_{\text{opt}}, \ell)}{C - 7},$$

denn bei jeder Anwendung verringert sich die Anzahl der Kreuzungen um mindestens $C - 3 - 4 = C - 7$. Bezeichnet $\alpha(\ell, i)$ die Anzahl der Anwendungen des Patching-Lemmas auf die Linie ℓ in der i-ten Iteration des obigen Vorgehens (beachte: $\alpha(\ell, i) = 0$ falls ℓ auf einer höheren Ebene liegt als der i-ten), dann kann man die Verlängerung der Steiner-Tour durch die Anwendung auf ℓ abschätzen durch

$$\sum_{i=1}^{N-1} \alpha(\ell, i) \cdot 6 \cdot 2^{N-i}.$$

Außerdem ist

$$\sum_{i=1}^{N-1} \alpha(\ell, i) \leq \frac{\mathrm{cr}(T_{\mathrm{opt}}, \ell)}{C-7},$$

so dass die erwartete Verlängerung bei der am Ende entstehenden Tour T höchstens

$$\begin{aligned}\sum_{\ell \in G_{N-1}} \sum_{j=1}^{N-1} p(\ell, j) \sum_{i \geq j} \alpha(\ell, i) \cdot 6 \cdot 2^{N-i} &= \sum_{\ell \in G_{N-1}} 6 \sum_{i=1}^{N-1} \alpha(\ell, i) \cdot 2^{N-i} \sum_{j=1}^{i} p(\ell, j) \\ &\leq \sum_{\ell \in G_{N-1}} 12 \sum_{i=1}^{N-1} \alpha(\ell, i) \leq \sum_{\ell \in G_{N-1}} 12 \, \frac{\mathrm{cr}(T_{\mathrm{opt}}, \ell)}{C-7}\end{aligned}$$

ist.

Nach Abschluss des Verfahrens oben wird jede Kante einer Region höchstens $C-4$ mal von T gekreuzt – wobei allerdings die zusätzlich entstehenden Kreuzungen nicht mitgerechnet sind. Man kann aber für jede Kante einer Region alle solche Kreuzungen entfernen, ohne die Tour zu verlängern, bis auf höchstens 4 (maximal zwei an jedem Ende der Kante), so dass am Ende tatsächlich eine leichte Steiner-Tour entsteht.

Schließlich ergibt sich (unter Verwendung des Kreuzungslemmas) für die erwartete Verlängerung E bei der konstruierten Tour gegenüber der Ausgangstour T_{opt}:

$$\begin{aligned}E \;&\leq \sum_{\ell \in G_{N-1}} \frac{N}{P} \, \mathrm{cr}(T_{\mathrm{opt}}, \ell) + \sum_{\ell \in G_{N-1}} 12 \, \frac{\mathrm{cr}(T_{\mathrm{opt}}, \ell)}{C-7} \\ &\leq c(T_{\mathrm{opt}}) \left(\frac{N}{P} + \frac{12}{C-7} \right) \leq c(T_{\mathrm{opt}}) \left(\frac{\varepsilon}{6} + \frac{\varepsilon}{3} \right) = c(T_{\mathrm{opt}}) \cdot \frac{\varepsilon}{2}.\end{aligned}$$

Aus der Markov-Ungleichung folgt nun: die Wahrscheinlichkeit, dass die Originaltour um höchstens $c(T_{\mathrm{opt}}) \cdot \varepsilon$ verlängert wird, ist mindestens $\frac{1}{\varepsilon}$. □

Algorithmus

Aroras Algorithmus berechnet nun eine optimale Steiner-Tour und wandelt diese durch Abkürzen in eine Tour durch die Punkte von V um. Das Verfahren funktioniert mittels dynamischer Programmierung, d. h. das zu lösende Gesamtproblem wird in Teilprobleme zerlegt, die rekursiv gelöst werden und deren Lösungen dann zusammengesetzt werden. Jedes Teilproblem ist dabei definiert durch

- eine Region R eines durch Linien in $G_i^{(a,b)}$, $1 \leq i \leq N-1$ definierten Gitters,
- eine Menge A von Portalen auf dem Rand von R, so dass $|A|$ gerade ist und höchstens C Portale aus A auf jeder Randkante von R liegen,
- ein perfektes Matching M im vollständigen Graphen mit Knotenmenge A. (Für unzusammenhängende Regionen R dürfen Kanten zwischen Portalen aus verschiedenen Komponenten nicht in M vorkommen.)

Eine Lösung eines solchen Teilproblems besteht aus einer Menge $\{P_e \mid e \in M\}$, wobei jedes $P_{\{v,w\}}$ ein Polygonzug in R zwischen den Portalen $v, w \in A$ ist, so dass jeder Punkt aus $V \cap R$ auf genau einem dieser Polygonzüge liegt. Eine Lösung ist optimal, wenn die Summe der Längen der Polygonzüge minimal ist.

Algorithmus (von Arora).
Eingabe: ε-diskretisiertes Euklidisches TSP $V \subset [0, 2^N]^2$ und eine Schranke $\varepsilon > 0$
Ausgabe: Tour durch V

1. **Initialisierung.**
   Wähle a und b zufällig gleichverteilt aus $\{0, 2, \ldots, 2^N - 2\}$.
   Setze $\mathcal{R}_0 := \{([0, 2^N]^2, V)\}$.

2. **Aufbau des Baums der Teilregionen.**
   Für $i = 1, \ldots, N-1$:

   2.1. Setze $\mathcal{R}_i := \emptyset$.

   2.2. Für jedes $(R, V_R) \in \mathcal{R}_{i-1}$ mit $V_R \geq 2$:
   Initialisiere die vier durch $G_i^{(a,b)}$ gebildeten Teilregionen $R_1, \ldots, R_4$ von R, setze $\mathcal{R}_i := \mathcal{R}_i \cup \{(R_1, V_R \cap R_1), \ldots, (R_4, V_R \cap R_4)\}$.

3. **Lösung der Teilprobleme.**
   Für $i = N-1, \ldots, 1$:
   Für jedes $(R, V_R) \in \mathcal{R}_i$ löse alle Teilprobleme bezüglich der Region R:

   3.1. Falls $|V_R| \leq 1$: Finde und speichere die optimale Lösung für jede mögliche Wahl von A und M durch Enumeration.

   3.2. Andernfalls: Setze für jede mögliche Wahl von A und M die Lösung aus den (bereits berechneten) optimalen Lösungen der Teilprobleme für die Teilregionen von R zusammen.

4. **Berechnung einer Gesamt-Steiner-Tour**
   Setze eine optimale leichte Steiner-Tour aus den Lösungen der vier Teilprobleme des Startquadrats zusammen.

5. **Berechnung einer Tour durch V**
   Berechne eine Tour durch V durch Weglassen der Portale. △

Satz. *Aroras Algorithmus berechnet eine Tour durch V, die mit Wahrscheinlichkeit mindestens $1/2$ eine Gütegarantie von $1 + \varepsilon$ hat. Die Laufzeit ist in $O(n(\log n)^c)$, wobei $c \in O(1/\varepsilon)$ nicht von n abhängt.* △

Beweis. Die erste Aussage folgt direkt aus dem Satz von Arora. Es bleibt noch die Laufzeit abzuschätzen. Dazu müssen wir abschätzen wieviele Regionen insgesamt abgearbeitet werden müssen und wieviel Aufwand dazu in jeder Region betrieben werden muss.

Betrachte zunächst den Baum aller Regionen mit Wurzel $([0, 2^N]^2, V)$. Jeder Baumknoten $(R, V_R) \in \mathcal{R}_i$ hat entweder 0 oder 4 Kinder, die alle Elemente von $\mathcal{R}_{i+1}$ sind und die Tiefe des Baums ist höchstens $N - 1$ (d. h. er hat höchstens N Ebenen). Sei S die Menge der Baumknoten, die 4 Kinder haben, welche alle Blätter sind. Die zugehörigen Regionen von S sind paarweise disjunkt und enthalten jeweils mindestens 2 Punkte aus V. Daraus folgt $|S| \leq \frac{n}{2}$. Weiterhin ist jeder Baumknoten, der kein Blatt ist, ein Vorgängerknoten von mindestens einem Knoten aus S; demnach gibt es höchstens $N \cdot \frac{n}{2}$ innere Knoten und deshalb höchstens $\frac{5}{2}Nn$ Knoten insgesamt, also Regionen, für welche die Teilprobleme gelöst werden müssen.

Für eine gegebene Region ist die Anzahl der Teilprobleme kleiner als $(P+2)^{4C}(4C)!$, was aus

$$\sum_{i=0}^{P+1} \binom{C}{i} \leq (P+2)^C$$

und der Tatsache folgt, dass es weniger als $(4C)!$ verschiedene perfekte Matchings gibt (sehr grobe Abschätzung). Jedes Teilproblem, dessen Region einen Punkt enthält (also das in Schritt 3.1 abgearbeitet wird) kann in Zeit $O(C)$ gelöst werden. Für Schritt 3.2 müssen insgesamt $(P+2)^{4C}(8C)!$ Möglichkeiten, die Lösungen der Teilregionen zusammenzusetzen, ausgewertet werden, was dank der bereits gespeicherten Teillösungen jeweils in konstanter Zeit geht. Für jede Region beläuft sich also der Aufwand auf $O\big((P+2)^{8C}(4C)!(8C)!\big)$.

Alles in allem ergibt sich damit, wegen $N \in O(\log {}^{n}/_{\varepsilon})$, $C \in O({}^{1}/_{\varepsilon})$ und $P \in O({}^{N}/_{\varepsilon})$, eine Gesamtlaufzeit des Algorithmus in

$$O\Big(\tfrac{5}{2}Nn \cdot (P+2)^{8C}(8C)^{12C}\Big) = O\left(n(\log n)^{O(1/\varepsilon)} O({}^{1}/_{\varepsilon})^{O(1/\varepsilon)}\right) \qquad \square$$

Die Derandomisierung des Algorithmus erfolgt nun einfach durch Ausführen mit allen möglichen Wahlen von a und b. Dafür gibt es insgesamt $2^N \cdot 2^N = 2^{2N}$ Möglichkeiten, d. h. (wegen $N \in O(\log {}^{n}/_{\varepsilon})$) der (randomisierte) Algorithmus muss $O({}^{n^2}/_{\varepsilon^2})$ oft ausgeführt werden, so dass der derandomisierte Algorithmus polynomial bleibt.

Korollar. *Es gibt ein polynomiales Approximationsschema für Euklidische TSPs. Genauer gesagt: Für jedes Euklidische TSP und jedes $\varepsilon > 0$ kann eine Tour mit Gütegarantie $(1+\varepsilon)$ in Laufzeit $O(n^3(\log n)^c)$ (mit c unabhängig von n) berechnet werden.* $\triangle$

Literaturverzeichnis

S. Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.

B. Korte and J. Vygen. *Combinatorial Optimization – Theory and Algorithms.* Springer Verlag, 3rd edition, 2006.