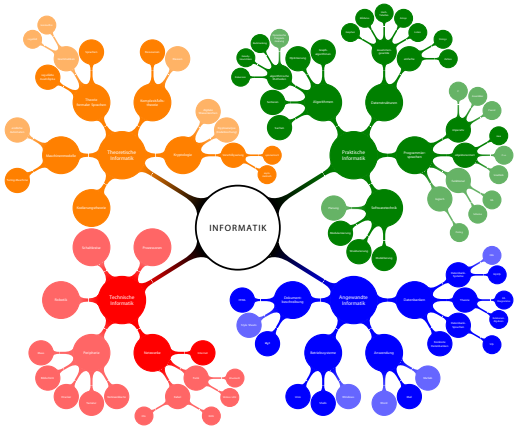
UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR
THEORETISCHE INFORMATIK

Kapitel 33

Approximationsalgorithmen

Wie faltet sich ein Protein

Vorlesung Einführung in die Informatik 1 vom 29. April 2014 von Till Tantau



Lernziele von Kapitel 33

1. Die Konzepte Optimierungsproblem, Lösung und Maße verstehen
2. Das Konzept der approximativen Lösung verstehen
3. Approximationsalgorithmen für das Handlungsreisenden-Problem und das Bin-Packing-Problem kennen

Gliederung von Kapitel 33

Optimierungsprobleme

Definition (Optimierungsprobleme)

Ein *Optimierungsproblem* ist ein Problem, bei dem es zu einer Eingabe

- viele mögliche *Lösungen* geben kann (eventuell auch keine),
- jede Lösung ein gewisses *Maß* hat (eine Zahl) und
- man eine Lösung sucht, deren Maß möglichst groß oder klein ist.

Ziele bei Optimierungsproblemen.

Sei ein Optimierungsproblem gegeben. Dann kann man mehrere Ziele haben (in absteigender Schwierigkeitsreihenfolge):

1. Für eine gegebene Eingabe möchte man eine *optimale Lösung* finden (also eine Lösung, deren Maß möglichst klein oder groß ist).
2. Für eine gegebene Eingabe möchte man eine *möglichst gute, aber vielleicht nicht optimale Lösung* finden.
3. Für eine gegebene Eingabe möchte man *überhaupt irgendeine Lösung finden*.
4. Für eine gegebene Eingabe möchte man *entscheiden, ob es eine Lösung gibt*.

Die Formalisierung von Optimierungsproblemen

Definition (Formales Optimierungsproblem)

Ein *Optimierungsproblem* besteht dann aus:

1. Einer Lösungsrelation S. Ist ein Paar (i, s) in dieser Menge, so ist s eine Lösung zu der Eingabe i.
2. Einer Maßfunktion. Dies ist eine Funktion $m : S \to \mathbb{N}$, die jeder Lösung ein Maß zuordnet.
3. Einem Typ. Dieser ist entweder »Maximierung« oder »Minimierung«.

Das Optimierungsproblem »Münzrückgabe«.

Das Optimierungsproblem

Eingabe	Ein (Multi-)Menge an Münzen und ein Wert w.
Lösungen	Teilmenge der Münzen, deren Summe w ist.
Maß	Anzahl der Münzen.
Ziel	Minimierung (man will möglichst wenig Kleingeld).

5-Minuten-Aufgabe

Zur Übung

Formulieren Sie das Bin-Packing-Problem als Optimierungsproblem, geben Sie also die Eingaben, die Lösungen, das Maß und das Ziel explizit an.

Gute versus schlechte Lösungen.

Definition

Sei P ein Optimierungsproblem. Dann ist

1. das *optimale Maß* zur Eingabe das minimale (oder maximale) Maß aller Lösungen zu x,
2. die *Güte* einer konkreten Lösung s zu x das Verhältnis

$$\frac{\text{Maß von } s}{\text{optimales Maß einer Lösung zu } x}.$$

Bei Maximierungsproblemen ist die Güte gerade der Kehrwert, so dass sie immer mindestens 1 ist.

Merke

Das *Maß* einer Lösung gibt ihre *»absoluten Kosten«* an. Die *Güte* einer Lösung gibt an, *um welchen Faktor* sie schlechter ist als eine optimale Lösung.

5-Minuten-Aufgabe

Zur Übung

Beim Münzrückgabeproblem sollen 1,86 Euro zurückgegeben werden, folgende Münzen stehen zur Verfügung :

Ein Algorithmus liefert als Lösung zweimal 50 Cent, dreimal 20 Cent, zweimal 10 Cent und dreimal 6 Cent. Berechnen sie das Maß und die Güte dieser Lösung.

Was tun, wenn man kein effizientes Lösungsverfahren kennt?

- Für viele Optimierungsprobleme *kennt man kein effizientes Lösungsverfahren*.
- Man kann dann probieren, eine *Heuristik* für das Problem zu finden.
- Eine solche liefert *eine Lösung* für das Problem, die im Allgemeinen recht gut ist (ihr Maß ist nahe am Optimum und ihre Güte damit nahe bei 1).
- Eine Heuristik *kann* aber auch (hoffentlich selten) sehr schlechte oder gar keine Lösungen liefern.

Beispiel

Optimierungsprobleme, für die man nur Heuristiken kennt:

- Handlungsreisenden-Problem,
- Protein-Faltung,
- DNA-Fragment-Assembly.

Approximationen – die besseren Heuristiken.

- Bei allgemeinen Heuristiken ist es unschön, dass sie sehr schlechte Lösungen liefern können.
- *Approximationsverfahren* sind Heuristiken, die eine *garantierte Güte* haben.
- Hat ein Verfahren beispielsweise Güte 3, so bedeutet dies, dass das Verfahren immer *Lösungen liefert, deren Maß höchstens dreimal so groß ist wie das Maß des Optimums.*

Beispiel

Optimierungsprobleme, die sich approximieren lassen:

- Handlungsreisenden-Problem,
- Bin-Packing.

Definition von Approximationsalgorithmen.

Definition

Sei P ein Optimierungsproblem und $r > 1$ eine rationale Zahl. Ein *r-Approximationsalgorithmus für P* ist ein Algorithmus, der

- zu jeder Eingabe x, zu der es eine Lösung gibt,
- eine Lösung ausrechnet, deren Güte kleiner oder gleich r ist.

Zur Erinnerung: Das Handlungsreisenden-Problem

Euklidisches Handlungsreisenden-Problem

Eingaben Ein Menge von Punkten in der Ebene.

Lösungen Rundreise (Folge der Punkte, die jeden Punkt genau einmal enthält).

Maß Summe der Länge der Strecken entlang der Rundreise.

Ziel Minimierung.

Allgemeines Handlungsreisenden-Problem

Eingaben Ein kantengewichteter Graph.

Lösungen Rundreise (Folge von miteinander verbundenen Knoten, die jeden Knoten genau einmal enthält).

Maß Summe der Gewichte der Kanten entlang der Rundreise.

Ziel Minimierung.

Ein Approximationsalgorithmus für das euklidische Handlungsreisenden-Problem.

- Man kennt *keinen effizienten Algorithmus* für das Handlungsreisenden-Problem.
- Man kennt aber viele gute *Approximations-Algorithmen*.

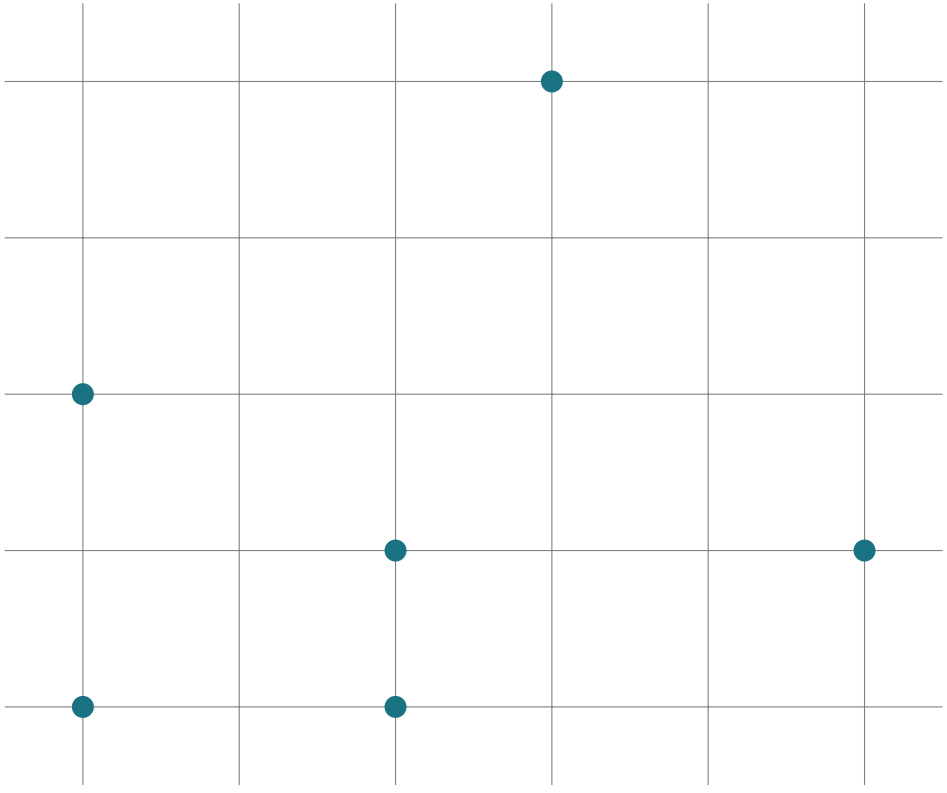
Algorithmus für das euklidische Handlungsreisenden-Problem

Eingabe ist eine Menge von Städten.

1. Erzeuge einen vollständigen Graphen, dessen Knoten Städte sind und dessen Kanten mit den Entfernungen der Städte gewichtet sind.
2. Berechne ein minimales Gerüst des Graphen.
3. Erzeuge eine Eulertour aus dem Gerüst.
4. Verkürze die Eulertour, bis jeder Knoten nur einmal besucht wird.

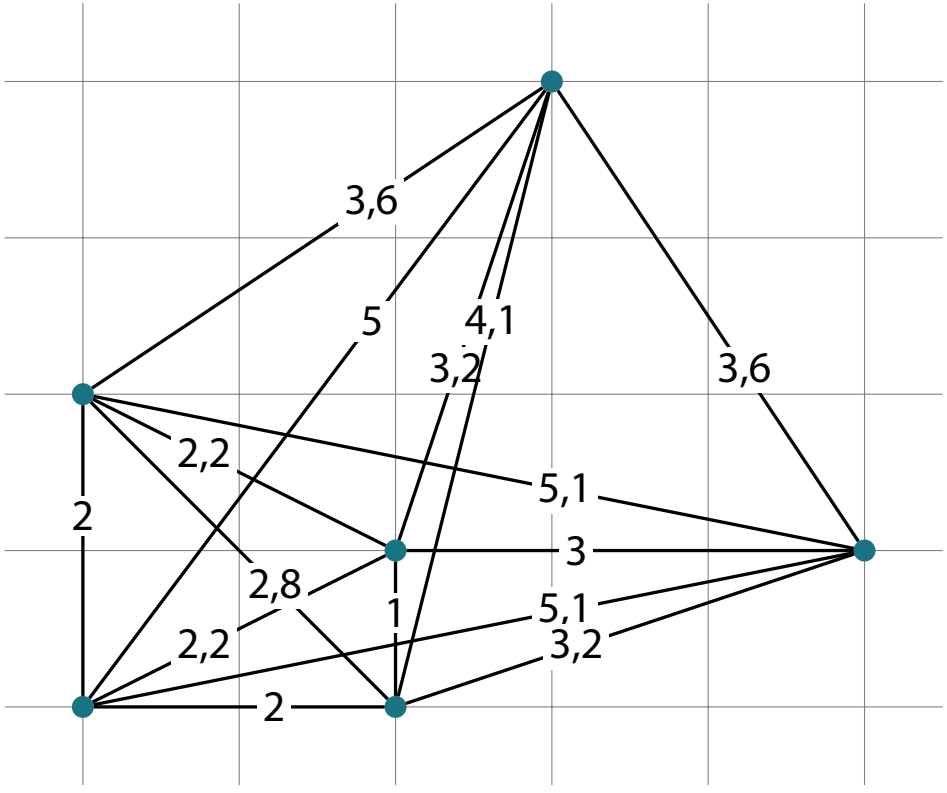
Der Algorithmus an einem Beispiel.

Schritt 0: Die Eingabe.



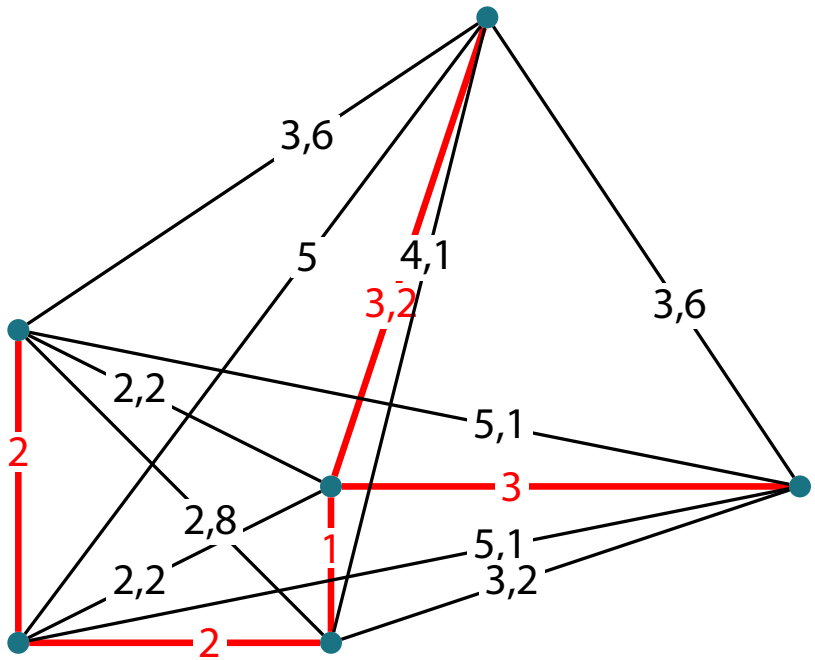
Der Algorithmus an einem Beispiel.

Schritt 1: Der Distanzgraph.



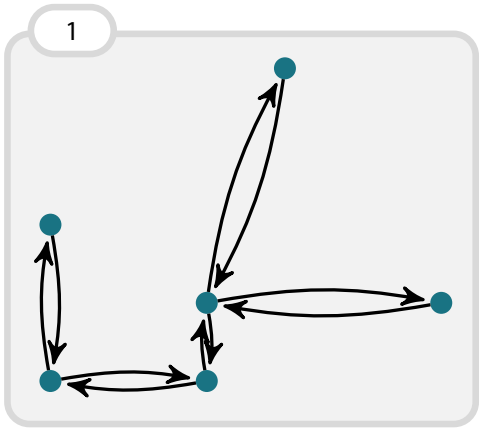
Der Algorithmus an einem Beispiel.

Schritt 2: Das Gerüst.



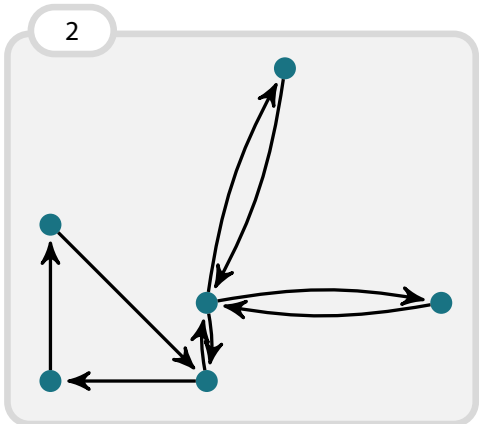
Der Algorithmus an einem Beispiel.

Schritte 3 und 4: Geradeziehen der Eulertour.



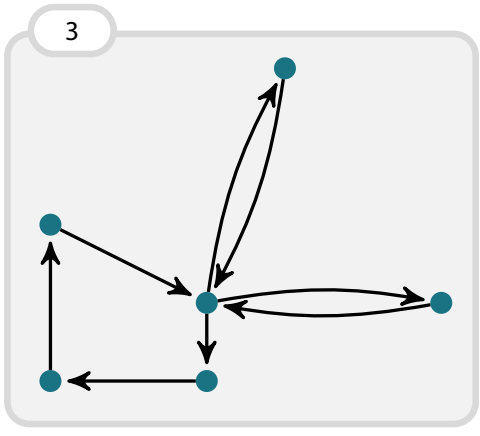
Der Algorithmus an einem Beispiel.

Schritte 3 und 4: Geradeziehen der Eulertour.



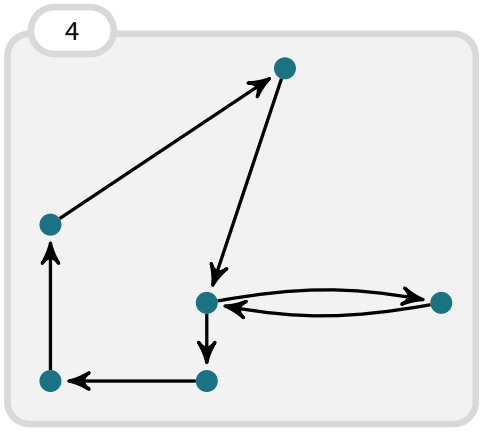
Der Algorithmus an einem Beispiel.

Schritte 3 und 4: Geradeziehen der Eulertour.



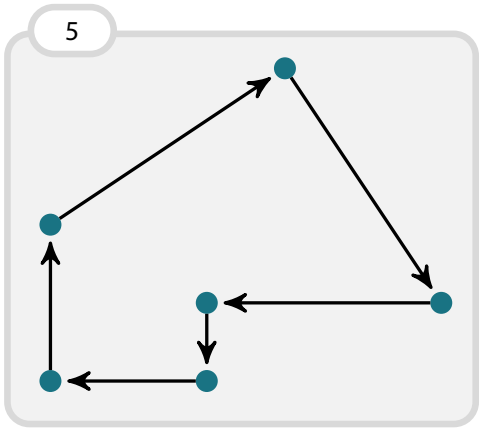
Der Algorithmus an einem Beispiel.

Schritte 3 und 4: Geradeziehen der Eulertour.



Der Algorithmus an einem Beispiel.

Schritte 3 und 4: Geradeziehen der Eulertour.



Der Algorithmus liefert eine 2-Approximation.

Satz

Der Algorithmus liefert immer eine Rundreise, die höchstens doppelt so lang ist wie die kürzeste.

Beweis.

1. Die kürzeste Rundreise habe die Länge x.
2. Löschen wir daraus eine Kante, so erhalten wir einen Pfad, der noch kürzer ist.
3. Dieser Pfad ist ein Gerüst. Also ist das Gewicht y des minimalen Gerüsts noch kleiner: $y < x$.
4. Die Eulertour hat Länge $2y < 2x$.
5. Die ausgegebene Tour ist kürzer als die Eulertour, also kürzer als $2x$. □

Das Bin-Packing-Problem

Das formale Bin-Packing-Problem.

Eingabe Eine Liste von Objektgrößen $(g_1, \ldots, g_n)$ und eine Eimergröße b.

Ausgaben Zuordnung von Objekten zu Eimern, so dass die Summe der Größen aller Objekte, die demselben Eimer zugeordnet sind, maximal b ist.

Maß Anzahl der benutzten Eimer.

Ziel Minimierung.

Wiederholung: Ein Greedy-Algorithmus für Bin-Packing.

Der First-Fit-Algorithmus

Für jeden Gegenstand tue folgendes:

1. Finde, von links beginnend, den ersten Eimer, in den der Gegenstand noch passt.
2. Platziere den Gegenstand in diesen Eimer.

Wie gut ist First-Fit?

Satz

Der First-Fit-Algorithmus ist ein 2-Approximationsalgorithmus für Bin-Packing.

Beweis.

- Betrachten wir eine Lösung, die First-Fit produziert hat.
- Dann *passt der Inhalt von je zwei* nebeneinander stehenden Eimer *nicht* in einen einzigen Eimer (sonst hätte First-Fit das nämlich getan).
- Also muss auch in einer optimalen Lösung für je zwei von First-Fit benutzte Eimer mindestens ein Eimer genutzt werden.

□

Zusammenfassung

1. (Formale) *Optimierungsprobleme* bestehen aus einer Lösungsrelation, einer Maßfunktion sowie einem Typ.
2. Ein *Approximationsverfahren* ist eine Heuristik, die Lösungen einer *garantierten Güte* liefert.
3. Für das *Handlungsreisenden-Problem* gibt es ein 2-Approximationsverfahren.
4. Der First-Fit-Algorithmus für das *Bin-Packing-Problem* ist ein 2-Approximationsverfahren.