

Kap. 7.1 Heuristiken

Kap. 7.2 Approximative Algorithmen
und Gütegarantien

Petra Mutzel

Lehrstuhl für Algorithm Engineering, LS11

23. VO

29. Juni 2006

Überblick

- Greedy-Heuristiken für TSP, Bin Packing, Knapsack
 - Hausaufgabenbesprechung: „schlechte Instanzen finden“
-
- Approximative Algorithmen
 - Gütegarantien für NN-Heuristik (TSP), FF-Heuristik (Bin Packing), Greedy-Heuristik (Knapsack)
 - Approximative Algorithmen mit Gütegarantie für TSP

Motivation

„Warum soll ich heute hier bleiben?“

Optimierung ist Klasse!

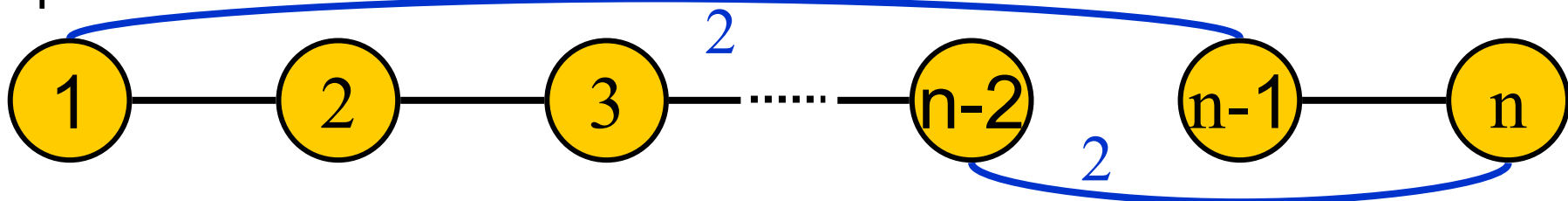
Wie gut ist die Nearest-Neighbor Heuristik (NN)?

- Kann beliebig schlechte Ergebnisse liefern,
- z.B. für $c(i,i+1)=1$ für $i=1,\dots,n-1$ und $c(n,1)=M$ (für eine sehr große Zahl M) und $c(i,j)=2$ sonst

NN-Lösung:



Optimale Tour:



Lösungswert NN: $(n-1)+M \gg$ Optimalwert: $(n-2)+4=n+2$

Bin-Packing / Packen von Kisten

- **Geg.:** Gegenstände $1, \dots, N$ der Größe w_i und beliebig viele Kisten der Größe K .
- **Gesucht:** Finde die kleinste Anzahl von Kisten, die alle Gegenstände aufnehmen.

First-Fit Heuristik: Jeder Gegenstand wird in die erstmögliche Kiste gelegt, in die er paßt.

Wie gut ist die First-Fit
Heuristik?

IHRE ERGEBNISSE 😊

First-Fit Heuristik für Bin-Packing

- Gegeben sind beliebig viele Kisten der Größe 101 und 37 Gegenstände der Größen:
7x Größe 6, 7x Größe 10, 3x Größe 16,
10x Größe 34, 10x Größe 51

Kiste 1: 7x Größe 6 5x Größe 10 Summe=92

Kiste 2: 2x Größe 10 3x Größe 16 Summe=68

Kisten 3-7: 2x Größe 34 Summe=68

Kisten 8-17: 1x Größe 51

FF-Lösung: 17 Kisten

Optimale Lösung: **Insgesamt: 10 Kisten**

Kisten 1-3: 1x Größe 51 1x Größe 34 1x Größe 16

Kisten 4-10: 1x Größe 51 1x Größe 34 1x Größe 10 1x Gr. 6

Das 0/1-Rucksackproblem

- **Geg.:** N Gegenstände mit Gewicht (Größe) w_i und Wert c_i , und ein Rucksack der Größe K .
- **Gesucht:** Menge der in den Rucksack gepackten Gegenstände mit maximalem Gesamtwert; dabei darf das Gesamtgewicht den Wert K nicht überschreiten.

Greedy-Heuristik an Beispiel für das 0/1-Rucksackproblem

K=17

Gegenstand	a	b	c	d	e	f	g	h
Gewicht	3	4	4	6	6	8	8	9
Wert	3	5	5	10	10	11	11	13
Nutzen	1	1,25	1,25	1,66	1,66	1,37	1,37	1,44

- Sortierung nach Nutzen := Wert c_i / Gewicht w_i ergibt: d,e,h,f,g,b,c,a
- Wir packen also in den Rucksack:

d (6) e (6) b(4)

Wert=25

Greedy-Heuristik für das 0/1-Rucksackproblem

- Sortiere die N Gegenstände nach ihrem Nutzen:= Wert c_i / Gewicht w_i
- Für alle Gegenstände i in der sortierten Reihenfolge:
 - Falls i noch in den Rucksack paßt: füge es hinzu.

Wie gut ist die
Greedy-Heuristik für das
0/1-Rucksackproblem?

IHRE VORSCHLÄGE 😊

Beispielinstanz für 0/1-Rucksackproblem

- N Gegenstände mit Gewichten und Werten $w_i = c_i = 1$ für $i=1, \dots, N-1$, $c_n = K-1$ und $w_n = K = MN$, wobei M eine sehr große Zahl ist; der Rucksack hat Größe K (also sehr groß).

- Greedy packt die ersten $N-1$ Gegenstände in den Rucksack; kein Platz mehr für N -tes Element. Greedy-Lösungswert: $N-1$

- Optimale Lösung packt nur N -tes Element ein; Lösungswert: $K-1 = MN-1$

Zusammenfassung

- **TSP:** NN-Heuristik: der berechnete Lösungswert kann beliebig weit vom optimalen Lösungswert entfernt sein.
- **Bin-Packing:** die von uns berechnete FF-Heuristik berechnete einen Lösungswert von 17, während der Optimalwert 10 war.
- **Rucksackproblem:** der von der Greedy-Heuristik berechnete Wert kann beliebig weit von der Optimallösung entfernt sein

Frage: Haben wir für Bin-Packing einfach keine „besonders schlechte“ Instanz gefunden? Oder ist Greedy hierfür immer relativ nah am Optimum?

Vertrauen in die Lösung

- Professor: Bei meiner Lösung entstehen Produktionskosten von 1.000.000 EUR.
- Praktiker: Gibt es keine bessere Lösung?
- Professor: Wir haben tagelang gerechnet und keine bessere gefunden.
- Praktiker: 😐 (Skepsis)
- Professor: Aber ich kann garantieren, dass es keine für weniger als 950.000 EUR geben kann.
- Praktiker: 😊 (höchstens 5% daneben!)

Motivation für Approximative Algorithmen

Kap. 7.2: Approximative Algorithmen und Gütegarantien

- **Approximative Algorithmen** sind Heuristiken, die (im vorhinein) eine Gütegarantie für die gefundene Lösung geben können.
- Z.B. der Art: „Die gefundene Lösung ist um höchstens $x\%$ schlechter als der Wert der optimalen Lösung.“

Approximative Algorithmen und Gütegarantien

- Sei A ein Algorithmus, der für jede Probleminstance P eines Optimierungsproblems Π eine zulässige Lösung mit positivem Wert liefert. Dann def. wir:
- $c_A(P)$ als den Wert der Lösung des Algorithmus A für Probleminstance $P \in \Pi$
- $c_{\text{opt}}(P)$ sei der optimale Wert für P .

- **Für Minimierungsprobleme gilt:**
- Falls $c_A(P) / c_{\text{opt}}(P) \leq \varepsilon$ für **alle** Probleminstance P und ein $\varepsilon > 0$, dann heißt A ein **ε -approximativer Algorithmus** und die Zahl ε heißt **Gütegarantie** von Algorithmus A .

Approximative Algorithmen und Gütegarantien

- Sei A ein Algorithmus, der für jede Probleminstance P eines Optimierungsproblems Π eine zulässige Lösung mit positivem Wert liefert. Dann def. wir:
- $c_A(P)$ als den Wert der Lösung des Algorithmus A für Probleminstance $P \in \Pi$
- $c_{\text{opt}}(P)$ sei der optimale Wert für P .

- Für **Max** imierungsprobleme gilt:
- Falls $c_A(P) / c_{\text{opt}}(P) \geq \varepsilon$ für **alle** Probleminstance P und ein $\varepsilon > 0$, dann heißt A ein **ε -approximativer Algorithmus** und die Zahl ε heißt **Gütegarantie** von Algorithmus A .

Approximative Algorithmen

- Für Minimierungsprobleme gilt: $\varepsilon \geq 1$
- Für Maximierungsprobleme gilt: $\varepsilon \leq 1$
- $\varepsilon = 1 \Leftrightarrow A$ ist exakter Algorithmus (berechnet immer den optimalen Wert)

- **Für Minimierungsprobleme gilt:**
- Falls $c_A(P) / c_{\text{opt}}(P) \leq \varepsilon$ für **alle** Probleminstanzen P und ein $\varepsilon > 0$, dann heißt A ein **ε -approximativer Algorithmus** und die Zahl ε heißt **Gütegarantie** von Algorithmus A .

Approximative Algorithmen

- Unsere Bin-Packing Beispielinstantz P:
- Lösung der FF-Heuristik: $c_A(P)=17$ Kisten
- Optimale Lösung: $c_{opt}(P)=10$ Kisten
- $c_A(P) / c_{opt}(P) = 17 / 10 = 1,7$
- FF-Heuristik ist vielleicht ein 1,7-approximativer Algorithmus (nur wenn $\leq 1,7$ für alle Instanzen gilt)
- ϵ kann auf keinen Fall kleiner als 1,7 sein

- **Für Minimierungsprobleme gilt:**
- Falls $c_A(P) / c_{opt}(P) \leq \epsilon$ für **alle** Probleminstanzen P und ein $\epsilon > 0$, dann heißt A ein **ϵ -approximativer Algorithmus** und die Zahl ϵ heißt **Gütegarantie** von Algorithmus A.

Approximative Algorithmen

- **Theorem:** Die First-Fit Heuristik besitzt asymptotisch eine Gütegarantie von 2.
- Es gilt: $c_{\text{FF}}(P) / c_{\text{opt}}(P) \leq 2 + 1/c_{\text{opt}}(P)$ für alle $P \in \Pi$

- **Beweis:** Offensichtlich gilt: Jede FF-Lösung füllt alle bis auf eine der belegten Kisten mindestens bis zur Hälfte (sonst hätten wir diese zusammenlegen können). Daraus folgt für alle $P \in \Pi$:

$$K/2 (c_{\text{FF}}(P) - 1) \leq \sum_{j=1..N} w_j \leq c_{\text{opt}}(P) K$$

Größe der verteilten
Gegenstände

Gesamtvolumen: in c_{opt}
Kisten paßt alles rein

Approximative Algorithmen

- **Theorem:** Die First-Fit Heuristik besitzt asymptotisch eine Gütegarantie von 2.
- Es gilt: $c_{\text{FF}}(P) / c_{\text{opt}}(P) \leq 2 + 1/c_{\text{opt}}(P)$ für alle $P \in \Pi$

- **Beweis:** Offensichtlich gilt: Jede FF-Lösung füllt alle bis auf eine der belegten Kisten mindestens bis zur Hälfte (sonst hätten wir diese zusammenlegen können). Daraus folgt für alle $P \in \Pi$:

$$K/2 (c_{\text{FF}}(P) - 1) \leq \sum_{j=1..N} w_j \leq c_{\text{opt}}(P) K$$

$$\Leftrightarrow c_{\text{FF}}(P) \leq 2 c_{\text{opt}}(P) + 1$$

Approximative Algorithmen

- **Theorem (ohne Beweis):** Die First-Fit Heuristik besitzt asymptotisch eine Gütegarantie von $17/10$.
- Es gilt: $c_{\text{FF}}(P) / c_{\text{opt}}(P) < 17/10 + 2/c_{\text{opt}}(P)$ für alle $P \in \Pi$

- Unsere Beispielinstantz zeigt auch, dass die FF-Heuristik keinen besseren Approximationsfaktor als $17/10$ besitzen kann.

Gütegarantie für Greedy-Heuristik für Rucksackproblem

Optimalwert: $MN-1$ \gg Lösungswert Greedy: $N-1$

- Unsere Beispielinstantz zeigt, dass die Greedy-Heuristik keinen besseren Approximationsfaktor als $(N-1) / (MN-1)$ besitzen kann.
- Da M beliebig groß sein kann, existiert kein festes $\epsilon > 0$ für die eine Güte festgelegt werden kann.
- Greedy-Heuristik für das Rucksackproblem kann also beliebig schlecht werden.

Eine kleine Änderung führt zu 2-approximativem Algorithmus

Gütegarantie für Nearest-Neighbor Heuristik für TSP

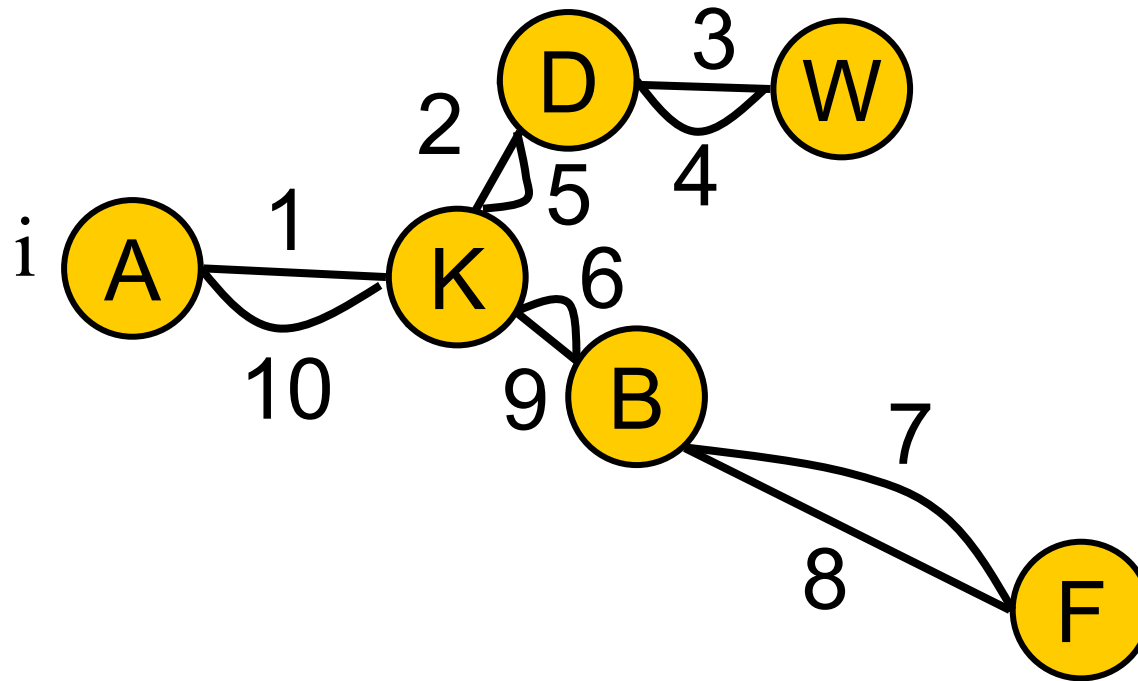
Lösungswert NN: $(n-1)+M \gg$ Optimalwert: $(n-2)+4=n+2$

- Unsere Beispielinstantz zeigt, dass die NN-Heuristik keinen besseren Approximationsfaktor als $(n-1+M) / (n+2)$ besitzen kann.
- Da M beliebig groß sein kann, existiert kein festes $\epsilon > 0$ für die eine Güte festgelegt werden kann.
- NN-Heuristik für das TSP kann also beliebig schlecht werden.

Spanning-Tree Heuristik für TSP

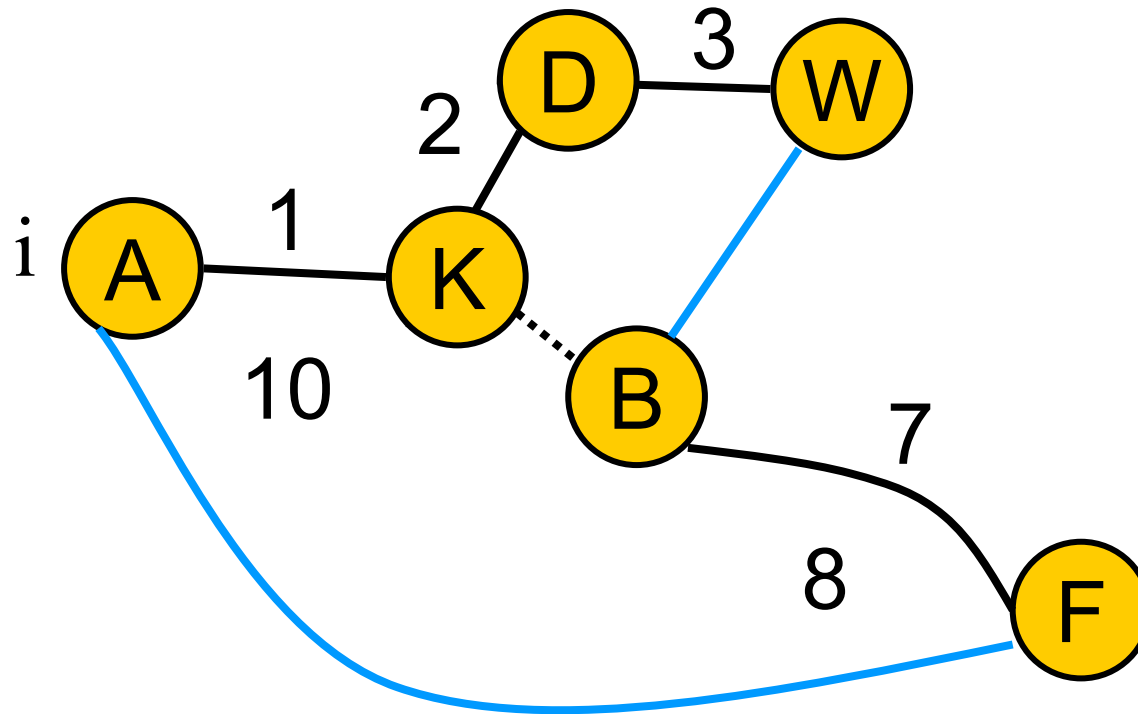
1. Bestimme einen MST B von G
2. Verdopple alle Kanten aus $B \rightarrow$ Graph $G_2 := (V, B_2)$
3. Bestimme eine Eulertour C in G_2 (Tour, die jede Kante genau einmal enthält); gib ihr Orientierung, wähle einen Knoten $i \in V$, markiere i , setze $p := i$ und $T := \emptyset$
4. Sind alle Knoten markiert, setze $T := T \cup \{(p, i)\} \rightarrow$ STOP; T ist Tour.
5. Laufe von p entlang der Orientierung von F bis ein unmarkierter Knoten q erreicht ist. Setze $T := T \cup \{(p, q)\}$, markiere q und gehe zu (4)

Beispiel für Spanning-Tree Heuristik



- MST mit verdoppelten Kanten
- Zahlen zeigen die Orientierung der Euler-Tour

Beispiel für Spanning-Tree Heuristik



Tour entsteht durch Wanderung entlang der Euler-Tour:
z.B. entsteht die Kante von W nach B, weil Knoten
D und K bereits markiert sind

Diskussion der Gütegarantie

- Auch für die Spanning-Tree Heuristik gibt es eine „schlechte“ Instanz, bei der Lösungen produziert werden, die beliebig weit vom optimalen Lösungswert entfernt sind.
- Man kann zeigen: Das Problem, das TSP-Problem für beliebiges $\epsilon > 1$ zu approximieren ist NP-schwierig.
- Aber: wenn man nur spezielle TSP-Instanzen betrachtet, dann kann man eine Gütegarantie finden.

Das Metrische TSP-Problem

- Ein TSP heißt **metrisch**, wenn für die gegebene Distanzmatrix alle $c_{ii}=0$ sind und die Dreiecksungleichung erfüllt ist, d.h. für alle Knoten i,j,k gilt: $c_{ik} \leq c_{ij} + c_{jk}$

- „Der direkte Weg von i nach k kann nicht länger sein als der Weg von i nach k über j .“

- TSP-Probleme aus der Praxis sind sehr oft metrisch (z.B. Wegeprobleme) --- sogar **euklidisch**: d.h. die Städte besitzen Koordinaten im 2-dim.Raum und die Distanzmatrix ist durch die euklidischen Distanzen gegeben.

Gütegarantie für ST-Heuristik

- Für metrische TSP und für die Spanning-Tree Heuristik gilt: $c_{ST}(P) / c_{opt}(P) \leq 2$ für alle $P \in \Pi$

- Beweis: $c_{ST}(P) \leq c_{B2}(P) = 2 \text{ MST}(P) \leq 2 c_{opt}(P)$

wegen
Dreiecksungleichung:
ST-Lösung läuft direkt

denn: in TSP-Lösung müssen
u.a. alle Knoten miteinander
verbunden sein;
der billigste Weg hierfür ist MST

Heuristik mit besserer Gütegarantie für TSP

nächstes Mal!