



Theoretische Grundlagen der Informatik

Vorlesung am 15.12.2021

Torsten Ueckerdt | 15. Dezember 2021

Relative Approximation: Genereller Ansatz

Bei Minimierungsproblemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\mathcal{A}(I)}{\text{OPT}(I)} \leq K$, also $\mathcal{A}(I) \leq K \cdot \text{OPT}(I)$.
- Wir brauchen:
 - eine **obere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
 - eine **untere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

$$\mathcal{A}(I) \leq X \text{ und } \text{OPT}(I) \geq Y \implies \mathcal{A}(I) \leq X = \frac{X \cdot Y}{Y} \leq \frac{X}{Y} \cdot \text{OPT}(I)$$

Relative Approximation: Genereller Ansatz

Bei Minimierungsproblemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\mathcal{A}(I)}{\text{OPT}(I)} \leq K$, also $\mathcal{A}(I) \leq K \cdot \text{OPT}(I)$.
- Wir brauchen:
 - eine **obere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
 - eine **untere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

$$\mathcal{A}(I) \leq X \text{ und } \text{OPT}(I) \geq Y \implies \mathcal{A}(I) \leq X = \frac{X \cdot Y}{Y} \leq \frac{X}{Y} \cdot \text{OPT}(I)$$

Bei Maximierungsproblemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\text{OPT}(I)}{\mathcal{A}(I)} \leq K$, also $\mathcal{A}(I) \geq \frac{1}{K} \cdot \text{OPT}(I)$.
- Wir brauchen:
 - eine **untere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
 - eine **obere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

$$\mathcal{A}(I) \geq X \text{ und } \text{OPT}(I) \leq Y \implies \mathcal{A}(I) \geq X = \frac{X \cdot Y}{Y} \geq \frac{X}{Y} \cdot \text{OPT}(I)$$

Beispiel: Greedy-Algorithmus für KNAPSACK

Idee:

Bevorzuge Elemente mit günstigem **Kosten-pro-Gewicht** Verhältnis, also hoher **Kostendichte**.

↪ Es werden der Reihe nach so viele Elemente wie möglich mit absteigender Kostendichte in die Lösung aufgenommen.

- Berechne die Kostendichten $p_i := \frac{c_i}{w_i}$ für $i = 1, \dots, n$
- Sortiere nach Kostendichten und indiziere: $p_1 \geq p_2 \geq \dots \geq p_n$
- Dies kann in Zeit $O(n \log n)$ geschehen.
- Für $i = 1$ bis n setze $x_i := \left\lfloor \frac{W}{w_i} \right\rfloor$ und $W := W - \left\lfloor \frac{W}{w_i} \right\rfloor \cdot w_i$.

Die Laufzeit dieses Algorithmus ist in $O(n \log n)$.

KNAPSACK-Instanz

$$M = \{1, \dots, n\},$$

Kosten c_1, \dots, c_n ,

Gewichte w_1, \dots, w_n ,

Gesamtgewicht W .

Beispiel: Greedy-Algorithmus für KNAPSACK

- Berechne die Kostendichten $p_i := \frac{c_i}{w_i}$ für $i = 1, \dots, n$
- Sortiere nach Kostendichten und indiziere: $p_1 \geq p_2 \geq \dots \geq p_n$
- Dies kann in Zeit $O(n \log n)$ geschehen.
- Für $i = 1$ bis n setze $x_i := \left\lfloor \frac{W}{w_i} \right\rfloor$ und $W := W - \left\lfloor \frac{W}{w_i} \right\rfloor \cdot w_i$.

KNAPSACK-Instanz

$M = \{1, \dots, n\}$,

Kosten c_1, \dots, c_n ,

Gewichte w_1, \dots, w_n ,

Gesamtgewicht W .

Satz.

Der Greedy-Algorithmus \mathcal{A} erfüllt $\mathcal{R}_{\mathcal{A}}(I) \leq 2$ für alle Instanzen I .

Bei Maximierungsproblemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\text{OPT}(I)}{\mathcal{A}(I)} \leq K$, also $\mathcal{A}(I) \geq \frac{1}{K} \cdot \text{OPT}(I)$.
- Wir brauchen:
 - eine **untere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
 - eine **obere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

Beispiel: Greedy-Algorithmus für KNAPSACK

- Berechne die Kostendichten $p_i := \frac{c_i}{w_i}$ für $i = 1, \dots, n$
- Sortiere nach Kostendichten und indiziere: $p_1 \geq p_2 \geq \dots \geq p_n$
- Dies kann in Zeit $O(n \log n)$ geschehen.
- Für $i = 1$ bis n setze $x_i := \left\lfloor \frac{W}{w_i} \right\rfloor$ und $W := W - \left\lfloor \frac{W}{w_i} \right\rfloor \cdot w_i$.

KNAPSACK-Instanz

$M = \{1, \dots, n\}$,

Kosten c_1, \dots, c_n ,

Gewichte w_1, \dots, w_n ,

Gesamtgewicht W .

Beweis:

(O.B.d.A. sei $w_1 \leq W$.)

Eine **untere Schranke** für $\mathcal{A}(I)$: $\mathcal{A}(I) \geq c_1 \cdot x_1 = c_1 \cdot \left\lfloor \frac{W}{w_1} \right\rfloor$

Eine **obere Schranke** für $\text{OPT}(I)$: $\text{OPT}(I) \leq p_1 \cdot W = \frac{c_1}{w_1} \cdot W$

Beispiel: Greedy-Algorithmus für KNAPSACK

- Berechne die Kostendichten $p_i := \frac{c_i}{w_i}$ für $i = 1, \dots, n$
- Sortiere nach Kostendichten und indiziere: $p_1 \geq p_2 \geq \dots \geq p_n$
- Dies kann in Zeit $O(n \log n)$ geschehen.
- Für $i = 1$ bis n setze $x_i := \left\lfloor \frac{W}{w_i} \right\rfloor$ und $W := W - \left\lfloor \frac{W}{w_i} \right\rfloor \cdot w_i$.

KNAPSACK-Instanz

$M = \{1, \dots, n\}$,

Kosten c_1, \dots, c_n ,

Gewichte w_1, \dots, w_n ,

Gesamtgewicht W .

Beweis:

(O.B.d.A. sei $w_1 \leq W$.)

Eine **untere Schranke** für $\mathcal{A}(I)$: $\mathcal{A}(I) \geq c_1 \cdot x_1 = c_1 \cdot \left\lfloor \frac{W}{w_1} \right\rfloor$

Eine **obere Schranke** für $\text{OPT}(I)$: $\text{OPT}(I) \leq p_1 \cdot W = \frac{c_1}{w_1} \cdot W$

Dann gilt

$$\text{OPT}(I) \leq c_1 \cdot \frac{W}{w_1} \leq c_1 \cdot \left(\left\lfloor \frac{W}{w_1} \right\rfloor + 1 \right) \leq 2 \cdot c_1 \cdot \left\lfloor \frac{W}{w_1} \right\rfloor \leq 2 \cdot \mathcal{A}(I).$$

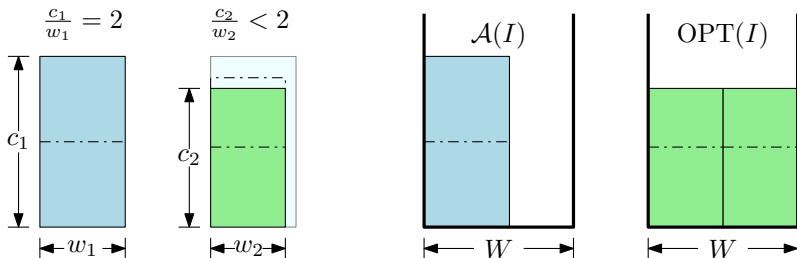
Also $\mathcal{R}_{\mathcal{A}}(I) \leq 2$.

Grenzen für den Greedy-Algorithmus

Bemerkung: Die Schranke $\mathcal{R}_{\mathcal{A}}(I) \leq 2$ ist in gewissem Sinne scharf.

- Sei $n = 2$, $w_2 = w_1 - 1$, $c_1 = 2 \cdot w_1$, $c_2 = 2 \cdot w_2 - 1$, $W = 2 \cdot w_2$.
- Dann ist $\frac{c_1}{w_1} = 2$ aber $\frac{c_2}{w_2} < 2$.
- Es gilt $\mathcal{A}(I) = c_1$ und $\text{OPT}(I) = 2c_2$, also

$$\frac{\text{OPT}(I)}{\mathcal{A}(I)} = \frac{2c_2}{c_1} = \frac{4w_1 - 6}{2w_1} \rightarrow 2 \quad \text{für } w_1 \rightarrow \infty$$



Mit \mathcal{NP} -schweren Problemen umgehen: Übersicht

Pseudopolynomiale Algorithmen

- Laufzeit: $\text{poly}(|I|, \max(I))$ polynomial in $|I|$ und $\max(I)$
- $\mathcal{A}(I) = \text{OPT}(I)$ optimal, kein Fehler

Bemerkung:

Es gibt noch weitere Ansätze.

Absolute Approximationsalgorithmen

- Laufzeit: $\text{poly}(|I|)$ polynomial
- $|\mathcal{A}(I) - \text{OPT}(I)| \leq K$ absoluter Fehler
 - $\mathcal{A}(I) \leq \text{OPT}(I) + K$ bei Minimierungsproblem
 - $\mathcal{A}(I) \geq \text{OPT}(I) - K$ bei Maximierungsproblem

Relative Approximationsalgorithmen

- Laufzeit: $\text{poly}(|I|)$ polynomial
- $\mathcal{A}(I) \leq K \cdot \text{OPT}(I)$ bei Minimierungsproblem relativer Fehler
- $\mathcal{A}(I) \geq \frac{1}{K} \cdot \text{OPT}(I)$ bei Maximierungsproblem

Bestmögliche Approximationsgüte

Definition.

Zu einem polynomialen Approximationsalgorithmus \mathcal{A} sei

$$\mathcal{R}_{\mathcal{A}}^{\infty} := \inf \left\{ r \geq 1 : \begin{array}{l} \text{es gibt ein } N_0 > 0, \text{ so dass } \mathcal{R}_{\mathcal{A}}(I) \leq r \\ \text{für alle } I \text{ mit } \text{OPT}(I) \geq N_0 \end{array} \right\}$$

Beispiel:

- Angenommen $\mathcal{A}(I) \leq K \cdot \text{OPT}(I) + 3$ für alle I .
- Dann ist $\mathcal{R}_{\mathcal{A}}(I) = K + 1$ für $\text{OPT}(I) = 3$
 und $\mathcal{R}_{\mathcal{A}}(I) = K + \frac{1}{2}$ für $\text{OPT}(I) = 6$, usw.
- Wir haben aber $\mathcal{R}_{\mathcal{A}}^{\infty} = K$.

Erinnerung:

$$\mathcal{R}_{\mathcal{A}}(I) = \frac{\mathcal{A}(I)}{\text{OPT}(I)}$$

für Minimierungsproblem

$$\mathcal{R}_{\mathcal{A}}(I) = \frac{\text{OPT}(I)}{\mathcal{A}(I)}$$

für Maximierungsproblem

Bestmögliche Approximationsgüte

Definition.

Zu einem polynomialen Approximationsalgorithmus \mathcal{A} sei

$$\mathcal{R}_{\mathcal{A}}^{\infty} := \inf \left\{ r \geq 1 : \begin{array}{l} \text{es gibt ein } N_0 > 0, \text{ so dass } \mathcal{R}_{\mathcal{A}}(I) \leq r \\ \text{für alle } I \text{ mit } \text{OPT}(I) \geq N_0 \end{array} \right\}$$

Beispiel:

- Angenommen $\mathcal{A}(I) \leq K \cdot \text{OPT}(I) + 3$ für alle I .
- Dann ist $\mathcal{R}_{\mathcal{A}}(I) = K + 1$ für $\text{OPT}(I) = 3$
und $\mathcal{R}_{\mathcal{A}}(I) = K + \frac{1}{2}$ für $\text{OPT}(I) = 6$, usw.
- Wir haben aber $\mathcal{R}_{\mathcal{A}}^{\infty} = K$.

$$\mathcal{A}(I) \leq \text{OPT}(I) + K \text{ für alle } I \quad \implies \quad \mathcal{R}_{\mathcal{A}}^{\infty} = 1$$

Erinnerung:

$$\mathcal{R}_{\mathcal{A}}(I) = \frac{\mathcal{A}(I)}{\text{OPT}(I)}$$

für Minimierungsproblem

$$\mathcal{R}_{\mathcal{A}}(I) = \frac{\text{OPT}(I)}{\mathcal{A}(I)}$$

für Maximierungsproblem

relative Approximation

$$\mathcal{A}(I) \leq K \cdot \text{OPT}(I) \text{ bzw. } \mathcal{R}_{\mathcal{A}}^{\infty} \leq K \\ \text{poly}(|I|)$$

absolute Approx.

$$|\mathcal{A}(I) - \text{OPT}(I)| \leq K \\ \text{poly}(|I|)$$

relative Approximation

$$\mathcal{A}(I) \leq K \cdot \text{OPT}(I) \text{ bzw. } \mathcal{R}_{\mathcal{A}}^{\infty} \leq K \\ \text{poly}(|I|)$$

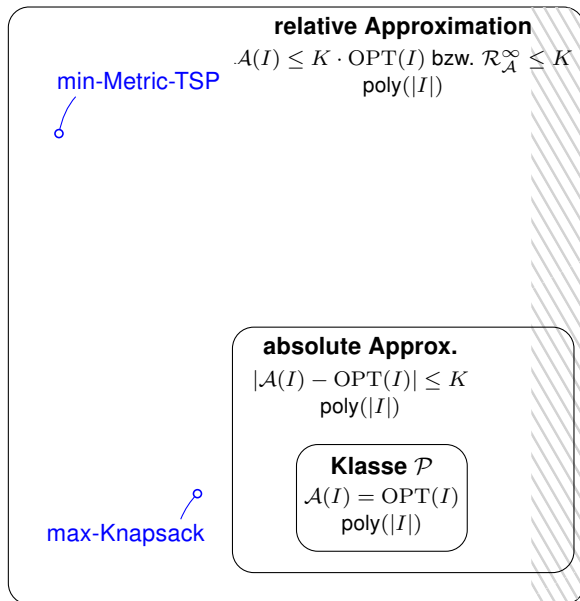
absolute Approx.

$$|\mathcal{A}(I) - \text{OPT}(I)| \leq K \\ \text{poly}(|I|)$$

Klasse \mathcal{P}

$$\mathcal{A}(I) = \text{OPT}(I) \\ \text{poly}(|I|)$$

Übersicht



Metrisches TSP

Optimalwertproblem min-METRIC-TSP

Gegeben: vollständiger Graph $G = (V, E)$,
Gewichtsfunktion $c: E \rightarrow \mathbb{Q}$
mit $c(u, w) \leq c(u, v) + c(v, w)$ für alle $u, v, w \in V$

Aufgabe: Minimiere die Länge bezüglich c von einer Tour zu G .

Satz.

Für das Optimalwertproblem min-METRIC-TSP existiert ein relativer Approximationsalgorithmus \mathcal{A} mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$.

Bemerkung:

- Es gilt sogar $\mathcal{R}_{\mathcal{A}}(I) \leq 2$ für alle Instanzen I .

2-Approximation von min-METRIC-TSP

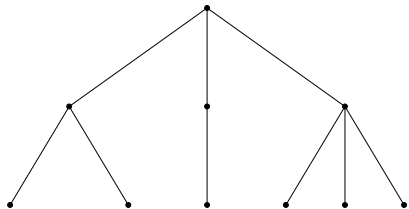
Beweis.

- Sei $I = (G = (V, E), c)$ eine Instanz von min-METRIC-TSP.

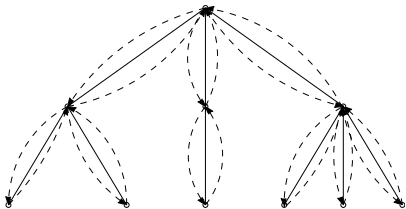
Betrachte folgenden Algorithmus \mathcal{A} :

- 1 Berechne einen *MST* (Minimum Spanning Tree) von G .
- 2 Wähle einen beliebigen Knoten w als Wurzel.
- 3 Durchlaufe den *MST* in einer Tiefensuche mit Startpunkt w
- 4 **Dies liefert:** Tour T mit Start- und Endpunkt w , die jede Kante genau zweimal durchläuft.
- 5 Konstruiere entlang T eine **abgekürzte Tour T'** , indem bereits besuchte Knoten übersprungen werden und die Tour T' beim nächsten unbesuchten Knoten fortgesetzt wird.
- 6 **Ergebnis:** $\mathcal{A}(I) = c(T') = \sum_{e \in T'} c(e)$

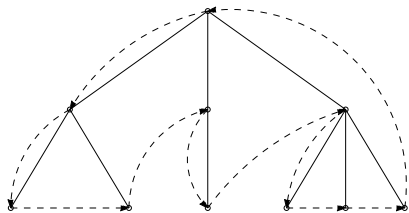
2-Approximation von min-METRIC-TSP



(a) MST eines Graphen



(b) Tiefensuch-Tour durch den MST



(c) TSP-Tour als abgekürzte Tiefensuch-Tour

2-Approximation von min-METRIC-TSP

Approximationsgüte:

Laufzeit:

- $O(n^2)$ für $n = |V|$.
- Das ist $\text{poly}(|I|)$.

2-Approximation von min-METRIC-TSP

Approximationsgüte:

Bei **Minimierungs**problemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\mathcal{A}(I)}{\text{OPT}(I)} \leq K$, also $\mathcal{A}(I) \leq K \cdot \text{OPT}(I)$.
- Wir brauchen:
 - eine **obere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
 - eine **untere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

Laufzeit:

- $O(n^2)$ für $n = |V|$.
- Das ist $\text{poly}(|I|)$.

2-Approximation von min-METRIC-TSP

Approximationsgüte:

Bei Minimierungsproblemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\mathcal{A}(I)}{\text{OPT}(I)} \leq K$, also $\mathcal{A}(I) \leq K \cdot \text{OPT}(I)$.

- Wir brauchen:

- eine **obere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
- eine **untere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

- **Obere Schranke** für $\mathcal{A}(I)$: $\mathcal{A}(I) = c(T') \leq c(T) = 2 \cdot c(\text{MST})$

- **Untere Schranke** für $\text{OPT}(I)$: $\text{OPT}(I) \geq c(\text{MST})$.

Denn: Eine TSP-Tour kann als ein aufspannender Baum plus eine zusätzliche Kante betrachtet werden. Und MST ist ein kürzester aufspannender Baum.

Laufzeit:

- $O(n^2)$ für $n = |V|$.
- Das ist $\text{poly}(|I|)$.

2-Approximation von min-METRIC-TSP

Approximationsgüte:

- **Obere Schranke** für $\mathcal{A}(I)$: $\mathcal{A}(I) = c(T') \leq c(T) = 2 \cdot c(MST)$
- **Untere Schranke** für $OPT(I)$: $OPT(I) \geq c(MST)$.

Denn: Eine TSP-Tour kann als ein aufspannender Baum plus eine zusätzliche Kante betrachtet werden. Und MST ist ein kürzester aufspannender Baum.

- Insgesamt erhält man

$$\mathcal{R}_{\mathcal{A}}(I) \leq \frac{\text{obere Schranke}}{\text{untere Schranke}} = \frac{2 \cdot c(MST)}{c(MST)} = 2.$$

Das heißt $\mathcal{A}(I) \leq 2 \cdot c(MST) \leq 2 \cdot OPT(I)$.

- $\rightsquigarrow \mathcal{R}_{\mathcal{A}}^{\infty} \leq 2.$

Laufzeit:

- $O(n^2)$ für $n = |V|$.
- Das ist $\text{poly}(|I|)$.

Bemerkungen zur Approximierbarkeit

min-METRIC-TSP

- Wir haben eine 2-Approximation mit $O(n^2)$ Laufzeit gesehen.
- **Christofides 1976:** Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- **Karpinski et al. 2015:** Es gibt **keine** $\frac{123}{122}$ -Approximation mit polynomialer Laufzeit.
- **Karlin et al. 2020:** Es gibt eine α -Approximation mit $\alpha < 1.5$ mit polynomialer Laufzeit.

Bemerkungen zur Approximierbarkeit

min-METRIC-TSP

- Wir haben eine 2-Approximation mit $O(n^2)$ Laufzeit gesehen.
- **Christofides 1976:** Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- **Karpinski et al. 2015:** Es gibt **keine** $\frac{123}{122}$ -Approximation mit polynomialer Laufzeit.
- **Karlin et al. 2020:** Es gibt eine α -Approximation mit $\alpha < 1.5$ mit polynomialer Laufzeit.

max-KNAPSACK

- Wir haben eine 2-Approximation mit $O(n \log n)$ Laufzeit gesehen.
 \rightsquigarrow Greedy Algorithmus
- Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.

Bemerkungen zur Approximierbarkeit

min-METRIC-TSP

- Wir haben eine 2-Approximation mit $O(n^2)$ Laufzeit gesehen.
- **Christofides 1976:** Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- **Karpinski et al. 2015:** Es gibt **keine** $\frac{123}{122}$ -Approximation mit polynomialer Laufzeit.
- **Karlin et al. 2020:** Es gibt eine α -Approximation mit $\alpha < 1.5$ mit polynomialer Laufzeit.

max-KNAPSACK

- Wir haben eine 2-Approximation mit $O(n \log n)$ Laufzeit gesehen.
 \rightsquigarrow Greedy Algorithmus
- Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- Es gibt eine 1.25-Approximation mit $O(n^3)$ Laufzeit.

Bemerkungen zur Approximierbarkeit

min-METRIC-TSP

- Wir haben eine 2-Approximation mit $O(n^2)$ Laufzeit gesehen.
- **Christofides 1976:** Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- **Karpinski et al. 2015:** Es gibt **keine** $\frac{123}{122}$ -Approximation mit polynomialer Laufzeit.
- **Karlin et al. 2020:** Es gibt eine α -Approximation mit $\alpha < 1.5$ mit polynomialer Laufzeit.

max-KNAPSACK

- Wir haben eine 2-Approximation mit $O(n \log n)$ Laufzeit gesehen.
 \rightsquigarrow Greedy Algorithmus
- Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- Es gibt eine 1.25-Approximation mit $O(n^3)$ Laufzeit.
- Es gibt eine 1.0001-Approximation mit $O(n^3)$ Laufzeit.

Bemerkungen zur Approximierbarkeit

min-METRIC-TSP

- Wir haben eine 2-Approximation mit $O(n^2)$ Laufzeit gesehen.
- **Christofides 1976:** Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- **Karpinski et al. 2015:** Es gibt **keine** $\frac{123}{122}$ -Approximation mit polynomialer Laufzeit.
- **Karlin et al. 2020:** Es gibt eine α -Approximation mit $\alpha < 1.5$ mit polynomialer Laufzeit.

max-KNAPSACK

- Wir haben eine 2-Approximation mit $O(n \log n)$ Laufzeit gesehen.
 \rightsquigarrow Greedy Algorithmus
- Es gibt eine 1.5-Approximation mit $O(n^3)$ Laufzeit.
- Es gibt eine 1.25-Approximation mit $O(n^3)$ Laufzeit.
- Es gibt eine 1.0001-Approximation mit $O(n^3)$ Laufzeit.

\rightsquigarrow **FPTAS**

Approximationsschemata

Definition.

Ein **Approximationsschema** für ein Optimierungsproblem Π ist eine Familie von Algorithmen $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$, so dass für alle $\varepsilon > 0$:

- $\mathcal{R}_{\mathcal{A}_\varepsilon} \leq 1 + \varepsilon$

Ein **PTAS** ist ein Approximationsschema bei dem

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

Ein **FPTAS** ist ein Approximationsschema bei dem

- die Laufzeit von \mathcal{A}_ε polynomial in $|I|$ und $\frac{1}{\varepsilon}$ ist.

(F)PTAS steht für
(Fully)
Polynomial
Time
Approximation
Scheme

Approximationsschemata

Definition.

Ein **Approximationsschema** für ein Optimierungsproblem Π ist eine Familie von Algorithmen $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$, so dass für alle $\varepsilon > 0$:

- $\mathcal{R}_{\mathcal{A}_\varepsilon} \leq 1 + \varepsilon$ \rightsquigarrow beliebig gute Approximation

Ein **PTAS** ist ein Approximationsschema bei dem

- die Laufzeit von \mathcal{A}_ε polynomial in $|I|$ ist. \rightsquigarrow $\text{poly}(|I|)$

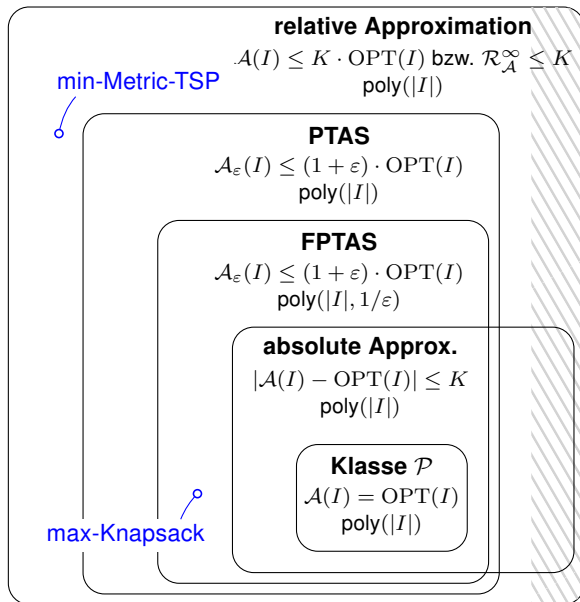
Ein **FPTAS** ist ein Approximationsschema bei dem

- die Laufzeit von \mathcal{A}_ε polynomial in $|I|$ und $\frac{1}{\varepsilon}$ ist. \rightsquigarrow $\text{poly}(|I|, 1/\varepsilon)$

- Ein **PTAS** erlaubt Laufzeiten von $O(n^{1/\varepsilon})$. $n = |I|$
z.B. $O(n)$ für 2-Approx., $O(n^2)$ für 1.5-Approx., $O(n^4)$ für 1.25-Approx.
- Ein **FPTAS** erlaubt Laufzeiten von $O(\frac{1}{\varepsilon} \cdot n)$.
z.B. $O(n)$ für 2-Approx., $O(n)$ für 1.5-Approx., $O(n)$ für 1.25-Approx.

(F)PTAS steht für
(Fully)
Polynomial
Time
Approximation
Scheme

Übersicht



Ein FPTAS für max-KNAPSACK

Optimierungsproblem max-KNAPSACK

Gegeben: Eine endliche Menge M ,
eine Gewichtsfunktion $w: M \rightarrow \mathbb{N}_0$,
eine Kostenfunktion $c: M \rightarrow \mathbb{N}_0$, $W \in \mathbb{N}$.

Aufgabe: Maximiere $c(M')$ für eine Teilmenge M' von M mit $w(M') \leq W$.

Ein FPTAS für max-KNAPSACK

Optimierungsproblem max-KNAPSACK

Gegeben: Eine endliche Menge M ,
 eine Gewichtsfunktion $w: M \rightarrow \mathbb{N}_0$,
 eine Kostenfunktion $c: M \rightarrow \mathbb{N}_0$, $W \in \mathbb{N}$.

Aufgabe: Maximiere $c(M')$ für eine Teilmenge M' von M mit $w(M') \leq W$.

Unser Vorgehen: Variere den pseudopolynomialen Algorithmus \mathcal{A} aus letzter Vorlesung.

\rightsquigarrow Laufzeit: $\mathcal{O}(|M| \cdot c(M))$

- Für $\varepsilon > 0$ entwerfe $(1 + \varepsilon)$ -Approximation \mathcal{A}_ε wie folgt:
 - ① Bei Eingabe $I = (M, w, c, W)$, berechne ein k aus $|I|$, $\max(I)$ und ε .
 - ② Skaliere Kostenfunktion $c'(i) = \lfloor c(i)/k \rfloor$.
 - ③ Berechne \mathcal{A} auf Eingabe (M, w, c', W) .
- Beweise: Laufzeit von $\mathcal{A}_\varepsilon = \text{poly}(|I|, \frac{1}{\varepsilon})$ und $\mathcal{R}_{\mathcal{A}_\varepsilon} \leq 1 + \varepsilon$.

Ein pseudopolynomialer, optimaler Algorithmus für max-KNAPSACK

Für $i \in M$, $r \leq c(M)$ berechne

$$w_r^i := \min\{w(M') \mid M' \subseteq \{1, \dots, i\}, c(M') = r\}.$$

■ Initialisierung

Für $i = 1, \dots, |M|$ setze $w_0^i := 0$

■ Berechnung

Für $r = 1, \dots, c(M)$ und $i = 1, \dots, |M|$ setze

$$w_r^i := \min \left\{ w_{r-c(i)}^{i-1} + w(i), w_r^{i-1} \right\}$$

■ Ausgabe $\mathcal{A}(I) := \max\{r \mid w_r^{|M|} \leq W\} = \text{OPT}(I)$

Ein pseudopolynomialer, optimaler Algorithmus für max-KNAPSACK

Für $i \in M$, $r \leq c(M)$ berechne

$$w_r^i := \min\{w(M') \mid M' \subseteq \{1, \dots, i\}, c(M') = r\}.$$

■ Initialisierung

Für $i = 1, \dots, |M|$ setze $w_0^i := 0$

■ Berechnung

Für $r = 1, \dots, c(M)$ und $i = 1, \dots, |M|$ setze

$$w_r^i := \min \left\{ w_{r-c(i)}^{i-1} + w(i), w_r^{i-1} \right\}$$

■ Ausgabe $\mathcal{A}(I) := \max\{r \mid w_r^{|M|} \leq W\} = \text{OPT}(I)$

↪ **Laufzeit:** in $O(|M| \cdot c(M))$.

↪ **Lösung:** optimal, d.h. $\mathcal{A}(I) = \text{OPT}(I)$.

↪ Optimaler pseudopolynomialer Algorithmus \mathcal{A} .

Ein FPTAS für max-KNAPSACK

- Bezeichne \mathcal{A} den vorigen pseudopolynomialen Algorithmus für KNAPSACK mit Laufzeit $O(|M| \cdot c(M))$.

Definiere Algorithmus \mathcal{A}_ε für $\varepsilon > 0$

- 1 Bei Eingabe $I = (M, w, c, W)$, berechne

$$c_{\max} := \max\{c(i) \mid i \in M\} \quad \text{und} \quad k := \frac{c_{\max}}{\left(\frac{1}{\varepsilon} + 1\right) \cdot |M|}$$

- 2 Betrachte die skalierte Instanz I_k mit $c'(i) := \left\lfloor \frac{c(i)}{k} \right\rfloor$ für alle $i \in M$.
- 3 Berechne \mathcal{A} mit Eingabe $I_k = (M, w, c', W)$.

Ein FPTAS für max-KNAPSACK

- Bezeichne \mathcal{A} den vorigen pseudopolynomialen Algorithmus für KNAPSACK mit Laufzeit $O(|M| \cdot c(M))$.

Definiere Algorithmus \mathcal{A}_ε für $\varepsilon > 0$

- 1 Bei Eingabe $I = (M, w, c, W)$, berechne

$$c_{\max} := \max\{c(i) \mid i \in M\} \quad \text{und} \quad k := \frac{c_{\max}}{\left(\frac{1}{\varepsilon} + 1\right) \cdot |M|}$$

- 2 Betrachte die skalierte Instanz I_k mit $c'(i) := \left\lfloor \frac{c(i)}{k} \right\rfloor$ für alle $i \in M$.
- 3 Berechne \mathcal{A} mit Eingabe $I_k = (M, w, c', W)$.

Satz.

$\mathcal{R}_{\mathcal{A}_\varepsilon}(I) \leq 1 + \varepsilon$ für alle $I \in D_{\text{II}}$ und die Laufzeit von \mathcal{A}_ε ist in $O(|I|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$, d.h. $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ist ein FPTAS für max-KNAPSACK.

Ein FPTAS für max-KNAPSACK

Satz.

$\mathcal{R}_{\mathcal{A}_\varepsilon}(I) \leq 1 + \varepsilon$ für alle $I \in D_{\Pi}$ und die Laufzeit von \mathcal{A}_ε ist in $O(|I|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$,
d.h. $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ist ein FPTAS für max-KNAPSACK.

Beweis:

Die Laufzeit von \mathcal{A}_ε ist $O(|M| \cdot c'(M))$ wobei

$$c'(M) = \sum_{i=1}^{|M|} \left\lfloor \frac{c(i)}{k} \right\rfloor \leq \sum_{i=1}^{|M|} \frac{c_i}{k} \leq |M| \cdot \frac{c_{\max}}{k} = \left(\frac{1}{\varepsilon} + 1 \right) |M|^2.$$

Also ist die Laufzeit von \mathcal{A}_ε in $O(|M|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$.

Ein FPTAS für max-KNAPSACK

Satz.

$\mathcal{R}_{\mathcal{A}_\varepsilon}(I) \leq 1 + \varepsilon$ für alle $I \in D_{\Pi}$ und die Laufzeit von \mathcal{A}_ε ist in $O(|I|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$,
 d.h. $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ist ein FPTAS für max-KNAPSACK.

▷ Für die Abschätzung $\text{OPT}(I) \leq (1 + \varepsilon) \cdot \mathcal{A}_\varepsilon(I)$.

Bei Maximierungsproblemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\text{OPT}(I)}{\mathcal{A}(I)} \leq K$, also $\mathcal{A}(I) \geq \frac{1}{K} \cdot \text{OPT}(I)$.
- Wir brauchen:
 - eine **untere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
 - eine **obere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

Ein FPTAS für max-KNAPSACK

Satz.

$\mathcal{R}_{\mathcal{A}_\varepsilon}(I) \leq 1 + \varepsilon$ für alle $I \in D_{\Pi}$ und die Laufzeit von \mathcal{A}_ε ist in $O(|I|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$,
 d.h. $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ist ein FPTAS für max-KNAPSACK.

▷ Für die Abschätzung $\text{OPT}(I) \leq (1 + \varepsilon) \cdot \mathcal{A}_\varepsilon(I)$.

Bei Maximierungsproblemen

- Wir wollen $\mathcal{R}_{\mathcal{A}}(I) = \frac{\text{OPT}(I)}{\mathcal{A}(I)} \leq K$, also $\mathcal{A}(I) \geq \frac{1}{K} \cdot \text{OPT}(I)$.
- Wir brauchen:
 - eine **untere Schranke** für $\mathcal{A}(I)$ “ \mathcal{A} ist gut”
 - eine **obere Schranke** für $\text{OPT}(I)$ “viel besser geht es nicht”

Wenn M^* optimal für I , also $\text{OPT}(I) = c(M^*)$, dann

$$\text{OPT}(I_k) \geq c'(M^*) = \sum_{i \in M^*} \left\lfloor \frac{c(i)}{k} \right\rfloor \geq \sum_{i \in M^*} \left(\frac{c(i)}{k} - 1 \right) \geq \frac{c(M^*)}{k} - |M|$$

Ein FPTAS für max-KNAPSACK

Satz.

$\mathcal{R}_{\mathcal{A}_\varepsilon}(I) \leq 1 + \varepsilon$ für alle $I \in D_{\Pi}$ und die Laufzeit von \mathcal{A}_ε ist in $O(|I|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$,
 d.h. $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ist ein FPTAS für max-KNAPSACK.

► Für die Abschätzung $\text{OPT}(I) \leq (1 + \varepsilon) \cdot \mathcal{A}_\varepsilon(I)$.

Wenn M^* optimal für I , also $\text{OPT}(I) = c(M^*)$, dann

$$\text{OPT}(I_k) \geq c'(M^*) = \sum_{i \in M^*} \left\lfloor \frac{c(i)}{k} \right\rfloor \geq \sum_{i \in M^*} \left(\frac{c(i)}{k} - 1 \right) \geq \frac{c(M^*)}{k} - |M|$$

- Eine **obere Schranke** für $\text{OPT}(I)$:

$$\mathcal{A}_\varepsilon(I) \geq k \cdot \mathcal{A}(I_k) = k \cdot \text{OPT}(I_k) \geq c(M^*) - k \cdot |M| = \text{OPT}(I) - k \cdot |M|$$

$$\text{Also } \boxed{\text{OPT}(I) \leq \mathcal{A}_\varepsilon(I) + k \cdot |M|}$$

Ein FPTAS für max-KNAPSACK

Satz.

$\mathcal{R}_{\mathcal{A}_\varepsilon}(I) \leq 1 + \varepsilon$ für alle $I \in D_{\Pi}$ und die Laufzeit von \mathcal{A}_ε ist in $O(|I|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$,
 d.h. $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ist ein FPTAS für max-KNAPSACK.

▷ Für die Abschätzung $\text{OPT}(I) \leq (1 + \varepsilon) \cdot \mathcal{A}_\varepsilon(I)$.

- Eine **obere Schranke** für $\text{OPT}(I)$:

$$\mathcal{A}_\varepsilon(I) \geq k \cdot \mathcal{A}(I_k) = k \cdot \text{OPT}(I_k) \geq c(M^*) - k \cdot |M| = \text{OPT}(I) - k \cdot |M|$$

$$\text{Also } \boxed{\text{OPT}(I) \leq \mathcal{A}_\varepsilon(I) + k \cdot |M|}$$

- Eine **untere Schranke** für $\mathcal{A}_\varepsilon(I)$:

$$\mathcal{A}_\varepsilon(I) \geq \text{OPT}(I) - k \cdot |M| \geq c_{\max} - k \cdot |M|$$

$$\text{Mit der Definition von } k \text{ also } \boxed{\mathcal{A}_\varepsilon(I) \geq k \cdot |M| \cdot (1/\varepsilon)}$$

Ein FPTAS für max-KNAPSACK

Satz.

$\mathcal{R}_{\mathcal{A}_\varepsilon}(I) \leq 1 + \varepsilon$ für alle $I \in D_{\Pi}$ und die Laufzeit von \mathcal{A}_ε ist in $O(|I|^3 \cdot \frac{1}{\varepsilon})$ für alle $\varepsilon > 0$,
 d.h. $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ist ein FPTAS für max-KNAPSACK.

▷ Für die Abschätzung $\text{OPT}(I) \leq (1 + \varepsilon) \cdot \mathcal{A}_\varepsilon(I)$.

- Eine **obere Schranke** für $\text{OPT}(I)$:

$$\mathcal{A}_\varepsilon(I) \geq k \cdot \mathcal{A}(I_k) = k \cdot \text{OPT}(I_k) \geq c(M^*) - k \cdot |M| = \text{OPT}(I) - k \cdot |M|$$

$$\text{Also } \boxed{\text{OPT}(I) \leq \mathcal{A}_\varepsilon(I) + k \cdot |M|}$$

- Eine **untere Schranke** für $\mathcal{A}_\varepsilon(I)$:

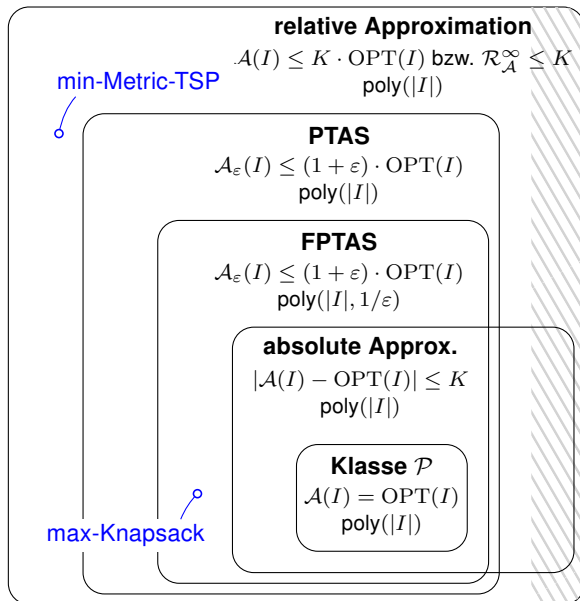
$$\mathcal{A}_\varepsilon(I) \geq \text{OPT}(I) - k \cdot |M| \geq c_{\max} - k \cdot |M|$$

$$\text{Mit der Definition von } k \text{ also } \boxed{\mathcal{A}_\varepsilon(I) \geq k \cdot |M| \cdot (1/\varepsilon)}$$

- Zusammen:

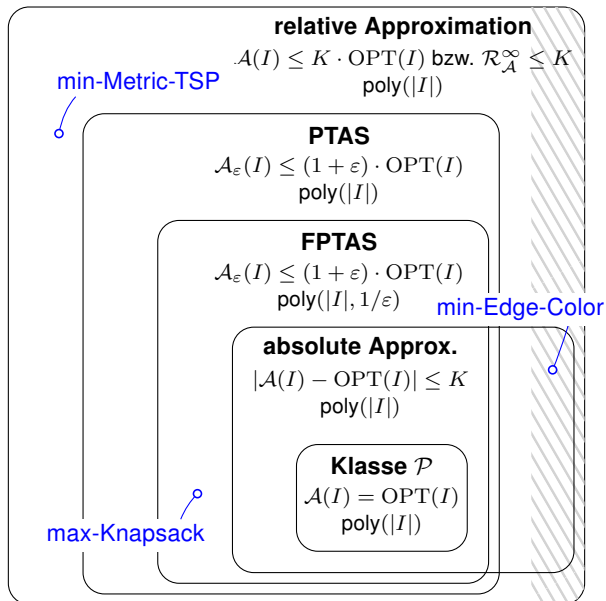
$$\text{OPT}(I) \leq \mathcal{A}_\varepsilon(I) + k \cdot |M| \leq \mathcal{A}_\varepsilon(I) + \varepsilon \cdot \mathcal{A}_\varepsilon(I) = (1 + \varepsilon) \cdot \mathcal{A}_\varepsilon(I)$$

Übersicht



Übersicht

min-Vertex-Color



min-VERTEX-COLOR und min-EDGE-COLOR

Optimierungsproblem min-VERTEX-COLOR

Gegeben: Graph $G = (V, E)$

Aufgabe: Färbe die Knoten in V mit möglichst wenig Farben, so dass je zwei adjazente Knoten verschiedene Farben besitzen.

Beide Entscheidungsprobleme “höchstens drei Farben” sind \mathcal{NP} -schwer.

Optimierungsproblem min-EDGE-COLOR

Gegeben: Graph $G = (V, E)$

Aufgabe: Färbe die **Kanten** in E mit möglichst wenig Farben, so dass je zwei adjazente **Kanten** verschiedene Farben besitzen.

Nicht-Existenz eines FPTAS

Optimierungsproblem min-VERTEX-COLOR

Gegeben: Graph $G = (V, E)$

Aufgabe: Färbe die Knoten in V mit möglichst wenig Farben, so dass je zwei adjazente Knoten verschiedene Farben besitzen.

Satz.

Sei Π ein \mathcal{NP} -schweres Optimierungsproblem mit

- $\text{OPT}(I) \in \mathbb{N}$ für alle $I \in D_{\Pi}$, und
- es existiert ein Polynom q mit $\text{OPT}(I) < q(|I|)$ für alle $I \in D_{\Pi}$.

Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es kein FPTAS $\{\mathcal{A}_{\varepsilon} \mid \varepsilon > 0\}$ für Π .

Nicht-Existenz eines FPTAS

Satz.

Sei Π ein \mathcal{NP} -schweres Optimierungsproblem mit

- $\text{OPT}(I) \in \mathbb{N}$ für alle $I \in D_{\Pi}$, und
- es existiert ein Polynom q mit $\text{OPT}(I) < q(|I|)$ für alle $I \in D_{\Pi}$.

Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es kein FPTAS $\{\mathcal{A}_{\varepsilon} \mid \varepsilon > 0\}$ für Π .

Beweis:

- O.B.d.A. sei Π ein Maximierungsproblem.
- Angenommen $\{\mathcal{A}_{\varepsilon} \mid \varepsilon > 0\}$ sei ein FPTAS für Π .
- Wir konstruieren optimalen, polynomialen Algorithmus \mathcal{A} für Π :
 - 1 Bei Eingabe $I \in D_{\Pi}$, berechne ein $\varepsilon_0 \leq \frac{1}{q(|I|)}$.
 - 2 Gebe $\mathcal{A}_{\varepsilon_0}(I)$ zurück. (Berechne Algorithmus $\mathcal{A}_{\varepsilon_0}$ auf Eingabe I .)

Laufzeit von $\mathcal{A}_{\varepsilon_0}$ ist $\text{poly}(|I|, \frac{1}{\varepsilon_0}) = \text{poly}(|I|)$, da $\frac{1}{\varepsilon_0} = q(|I|) = \text{poly}(|I|)$.

Nicht-Existenz eines FPTAS

Satz.

Sei Π ein \mathcal{NP} -schweres Optimierungsproblem mit

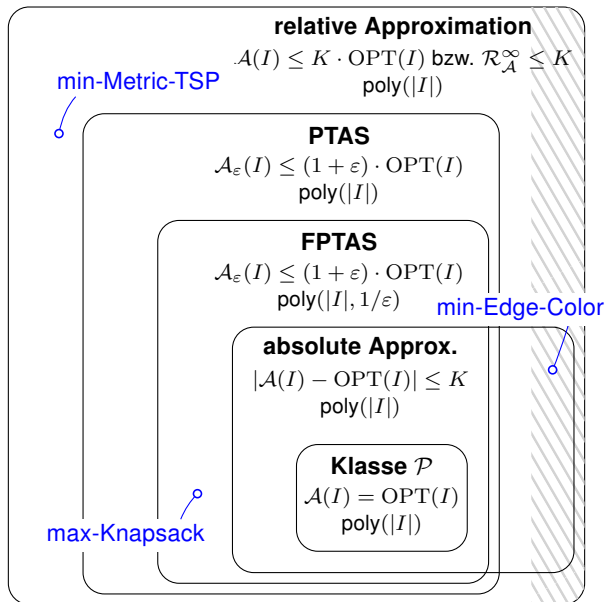
- $\text{OPT}(I) \in \mathbb{N}$ für alle $I \in D_{\Pi}$, und
- es existiert ein Polynom q mit $\text{OPT}(I) < q(|I|)$ für alle $I \in D_{\Pi}$.

Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es kein FPTAS $\{\mathcal{A}_{\varepsilon} \mid \varepsilon > 0\}$ für Π .

- Für die Güte beobachte: $\text{OPT}(I) \leq (1 + \varepsilon_0)\mathcal{A}_{\varepsilon_0}(I)$ und
 $\text{OPT}(I) < q(|I|) = \frac{1}{\varepsilon_0}$.
- Also gilt $0 \leq \text{OPT}(I) - \mathcal{A}_{\varepsilon_0}(I) \leq \varepsilon_0 \cdot \mathcal{A}_{\varepsilon_0}(I) \leq \varepsilon_0 \cdot \text{OPT}(I) < 1$.
- Da $\text{OPT}(I), \mathcal{A}_{\varepsilon_0}(I) \in \mathbb{N}$, ist $\text{OPT}(I) = \mathcal{A}_{\varepsilon_0}(I)$.
- Demnach ist $\mathcal{A}(I) = \mathcal{A}_{\varepsilon_0}(I) = \text{OPT}(I)$, also $\Pi \in \mathcal{P}$.
- Da Π \mathcal{NP} -schwer ist, folgt $\mathcal{P} = \mathcal{NP}$.

Übersicht

min-Vertex-Color



Approximierbarkeit von min-VERTEX-COLOR

Optimierungsproblem min-VERTEX-COLOR

Gegeben: Graph $G = (V, E)$

Aufgabe: Färbe die Knoten in V mit möglichst wenig Farben, so dass je zwei adjazente Knoten verschiedene Farben besitzen.

Satz.

Falls $\mathcal{P} \neq \mathcal{NP}$, dann existiert kein relativer Approximationsalgorithmus \mathcal{A} für min-VERTEX-COLOR mit $\mathcal{R}_{\mathcal{A}}^{\infty} < \frac{4}{3}$.

Approximierbarkeit von min-VERTEX-COLOR

Optimierungsproblem min-VERTEX-COLOR

Gegeben: Graph $G = (V, E)$

Aufgabe: Färbe die Knoten in V mit möglichst wenig Farben, so dass je zwei adjazente Knoten verschiedene Farben besitzen.

Satz.

Falls $\mathcal{P} \neq \mathcal{NP}$, dann existiert kein relativer Approximationsalgorithmus \mathcal{A} für min-VERTEX-COLOR mit $\mathcal{R}_{\mathcal{A}}^{\infty} < \frac{4}{3}$.

- Angenommen es gibt einen relativen Approximationsalgorithmus \mathcal{A} für min-VERTEX-COLOR mit $\mathcal{R}_{\mathcal{A}}^{\infty} < \frac{4}{3}$.
- Wir benutzen \mathcal{A} um Entscheidungsproblem 3COLOR zu lösen.
- Da 3COLOR \mathcal{NP} -schwer ist, folgt $\mathcal{P} = \mathcal{NP}$.

Approximierbarkeit von min-VERTEX-COLOR

Zu zwei Graphen $G_1 = (V_1, E_1)$ und $G_2 = (V_2, E_2)$ sei

$$G := (V, E) := G_1[G_2]$$

definiert durch

$$V := V_1 \times V_2$$

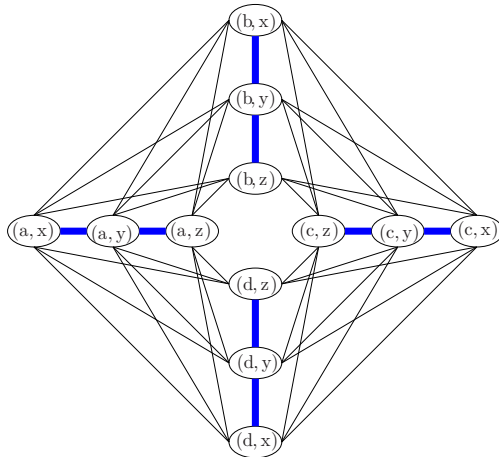
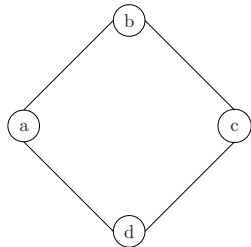
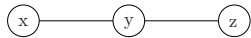
und

$$E := \left\{ \{(u_1, u_2), (v_1, v_2)\} \mid \begin{array}{l} \text{entweder } \{u_1, v_1\} \in E_1, \text{ oder} \\ u_1 = v_1 \text{ und } \{u_2, v_2\} \in E_2 \end{array} \right\}$$

Anschaulich

- Jeder Knoten aus G_1 wird durch eine Kopie von G_2 ersetzt
- Jede Kante aus E_1 durch einen vollständig bipartiten Graphen zwischen den entsprechenden Kopien.

Approximierbarkeit von min-VERTEX-COLOR



Approximierbarkeit von min-VERTEX-COLOR

$$\mathcal{R}_{\mathcal{A}}^{\infty} := \inf \left\{ r \geq 1 \mid \begin{array}{l} \text{es gibt ein } N_0 > 0, \text{ so dass } \mathcal{R}_{\mathcal{A}}(I) \leq r \\ \text{für alle } I \text{ mit } \text{OPT}(I) \geq N_0 \end{array} \right\}$$

- Angenommen es gibt einen relativen Approximationsalgorithmus \mathcal{A} für min-VERTEX-COLOR mit $\mathcal{R}_{\mathcal{A}}^{\infty} < \frac{4}{3}$.
- Dann existiert ein $N_0 \in \mathbb{N}$ so, dass $\mathcal{A}(G) < \frac{4}{3} \text{OPT}(G)$ für alle Graphen G mit $\text{OPT}(G) \geq N_0$.

Approximierbarkeit von min-VERTEX-COLOR

- Dann existiert ein $N_0 \in \mathbb{N}$ so, dass $\mathcal{A}(G) < \frac{4}{3} \text{OPT}(G)$ für alle Graphen G mit $\text{OPT}(G) \geq N_0$.
- Sei also $G = (V, E)$ eine beliebige Instanz von 3COLOR.
- Dann definiere $G^* := K_{N_0}[G]$, wobei K_{N_0} der vollständige Graph mit N_0 Knoten ist.
- Dann gilt: $\text{OPT}(G^*) = N_0 \cdot \text{OPT}(G) \geq N_0$.

Fallunterscheidung:

- Falls G Ja-Instanz (also dreifärbbar) ist, gilt:

$$\mathcal{A}(G^*) < \frac{4}{3} \text{OPT}(G^*) = \frac{4}{3} \cdot N_0 \cdot \text{OPT}(G) \leq \frac{4}{3} \cdot N_0 \cdot 3 = 4N_0.$$

- Andererseits, falls G Nein-Instanz (also nicht dreifärbbar) ist, gilt

$$\mathcal{A}(G^*) \geq \text{OPT}(G^*) = N_0 \cdot \text{OPT}(G) \geq 4N_0.$$

Fazit: G ist Ja-Instanz (dreifärbbar) genau dann, wenn $\mathcal{A}(G^*) < 4N_0$.

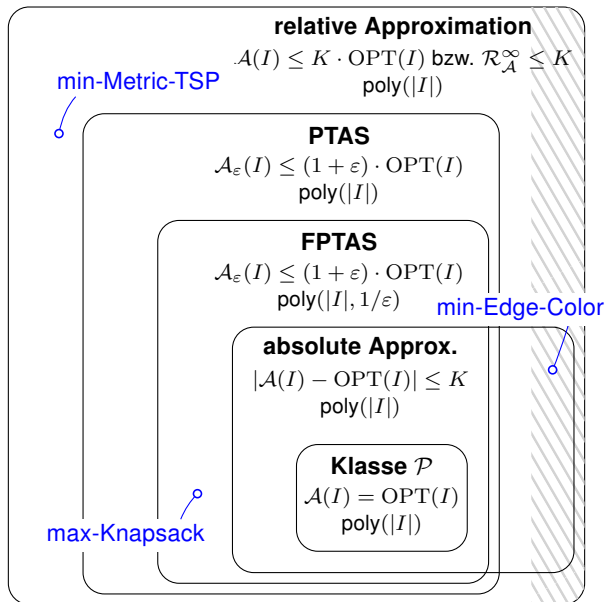
Approximierbarkeit von min-VERTEX-COLOR

Fazit: G ist Ja-Instanz (dreifärbbar) genau dann, wenn $\mathcal{A}(G^*) < 4N_0$.

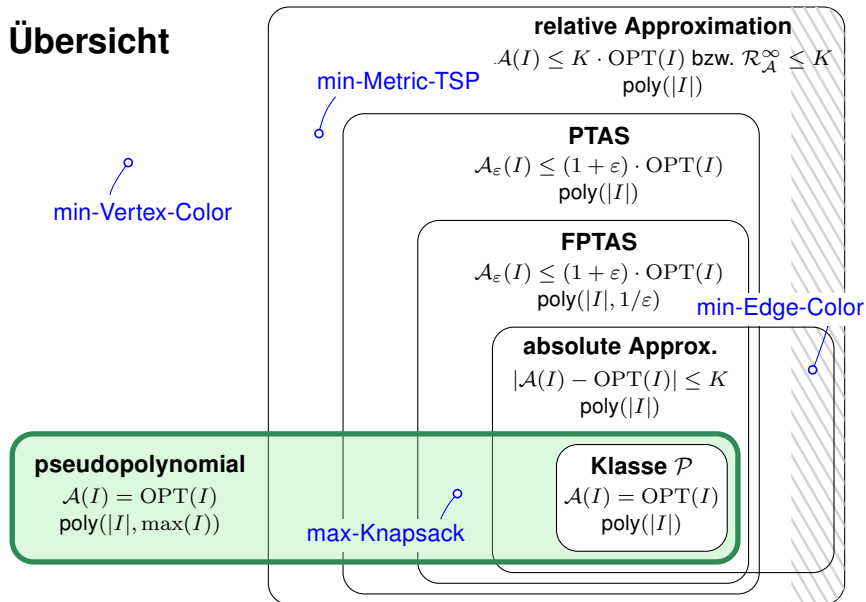
- Die Größe von G^* ist polynomial in der Größe von G .
- Also kann G^* in polynomialer Zeit konstruiert werden.
- Damit ist die Anwendung von \mathcal{A} auf G^* polynomial in der Größe von G .
- Also haben wir einen polynomialen Algorithmus zur Lösung von 3COLOR konstruiert.
- Da 3COLOR \mathcal{NP} -schwer ist, folgt damit dass $\mathcal{P} = \mathcal{NP}$.

Übersicht

min-Vertex-Color



Übersicht



Ein allgemeines Resultat

Satz.

Sei Π ein Optimierungsproblem für das gilt:

- $\text{OPT}(I) \in \mathbb{N}$ für alle $I \in D_{\Pi}$
- es existiert ein Polynom q mit $\text{OPT}(I) \leq q(|I| + \max(I))$

Falls Π ein FPTAS hat, so hat es einen pseudopolynomialen optimalen Algorithmus.

Ende des Kapitels

- Wir haben heute das Kapitel **Komplexitätstheorie** abgeschlossen.
- Wir werden aber nochmal über Turing-Maschinen sprechen.

Testen Sie sich:

Können Sie mit folgenden Begriffen etwas anfangen?

CLIQUE	polynomiale Transformation	3SAT	
Zeitkomplexitätsfunktion	Turing-Maschine	Approximation	
Optimierungsproblem	\mathcal{NP}	Orakel	Eingabekodierung
Instanz	pseudopolynomial	\mathcal{NP} -vollständig	(F)PTAS
Entscheidungsproblem	\mathcal{P}	Nichtdeterminismus	

Übersicht

