Näherungsalgorithmen

(Approximationsalgorithmen)

WiSe 2010/11 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

Näherungsalgorithmen Gesamtübersicht

- Organisatorisches
- Einführung / Motivation
- Grundtechniken für Näherungsalgorithmen
- Approximationsklassen (Approximationstheorie)

Approximationstheorie

- Absolute Approximation
- Relative Approximation: die Klasse APX
- Polynomzeit-Approximationsschemata PTAS
- Zwischen APX und NPO
- Zwischen PTAS und APX
- Approximationsklassen und Reduktionen

Zwischen PTAS und APX: eine Verallgemeinerung der Klasse PTAS

Definition Ein NPO-Problem $\mathcal{P}$ liegt in PTAS$^\infty$, in Worten: $\mathcal{P}$ hat ein *asymptotisches Approximationsschema*, wenn es einen Algorithmus $\mathcal{A}$ gibt und eine Konstante k, sodass (für jede Instanz x von $\mathcal{P}$ und für jede rationale Zahl $r \geq 1$ als Eingaben) $\mathcal{A}(x, r)$ in Polynomzeit eine Lösung liefert, deren Leistungsgüte höchstens $r + \frac{k}{m^*(x)}$ beträgt.

Natürlich gilt: PTAS $\subseteq$ PTAS$^\infty$ $\subseteq$ APX. $\quad (*)$

Mitteilung: Unter der Annahme P $\neq$ NP sind beide Inklusionen in $(*)$ echt.

Wir werden hier zwei Probleme aus PTAS$^\infty$ betrachten: das Kantenfärben in einem Graphen sowie Bin-Packing.

Kantenfärben

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

$S:$ Eine *Färbung der Kanten*, d.h. eine Partition von E in $E_1, \ldots, E_K$, sodass für jedes $1 \leq i \leq K$ gilt: Keine zwei Kanten aus E_i haben einen gemeinsamen Endpunkt

$m:$ Anzahl der Farben, d. h. also K.

opt : min.

Satz: (Vizing) Es gibt einen Polynomzeitalgorithmus, welcher bei Eingabe eines Graphen G mit Maximalgrad Δ eine Kantenfärbung mit höchstens $\Delta + 1$ vielen Farben liefert.

<u>Beweis:</u> Ist $G = (V, E)$ der Eingabegraph vom Maximalgrad Δ, so färbt $\mathcal{A}$ G, indem $\mathcal{A}$ nacheinander die Kanten von E färbt, wobei evtl. „frühere“ Kantenfärbungen später revidiert werden.

1. $E' := E; \overline{E} := \emptyset;$
2. Solange $E' \neq \emptyset$, tue:

   2a) Wähle Kante $\{u, v\} \in E'$.

   2b) Erweitere die Kantenfärbung von $\overline{E}$ auf $\overline{E} \cup \{\{u, v\}\}$, sodass $\left(V, \overline{E} \cup \{\{u, v\}\}\right)$ mit höchstens $\Delta + 1$ vielen Farben gefärbt ist.

   2c) $E' := E' \setminus \{\{u, v\}\}; \overline{E} := \overline{E} \cup \{\{u, v\}\}$

Präzisierung von 2b)

Farbmenge: $F = \{1, \ldots, \Delta + 1\}$.
Gesucht: Färbung: $f : E \to F$.
Annahme: Für $(V; \overline{E})$ ist eine Kantenfärbung mit höchstens $\Delta + 1$ vielen Farben konstruiert.
(Formal führen wir also einen Induktionsbeweis über $\left|\overline{E}\right|$ mit trivialen Induktionsanfang für $\overline{E} = \emptyset$.)
f ist also partiell und vorläufig auf $\overline{E}$ festgelegt.
Notation:

- Ist $v \in V$ beliebig, so bezeichne

  $$C(v) = \left\{\tilde{f} \in F \mid \neg\exists u \in V : \{u, v\} \in E \wedge f(\{u, v\}) = \tilde{f}\right\}$$

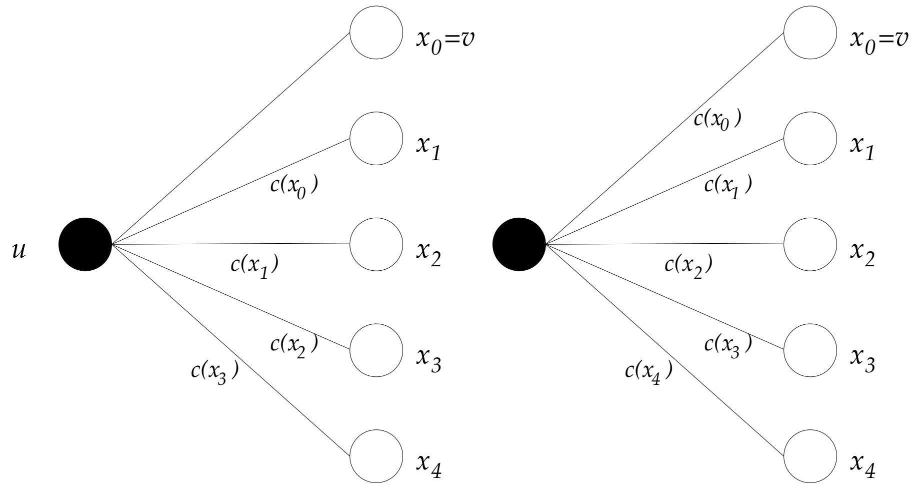
  die Farbmenge, die für „weitere Kanten“, die mit v inzidieren, noch „frei“ ist.

- Da $\delta(v) \leq \Delta$, ist für alle $v \in V$ die Menge $C(v) \neq \emptyset$. Daher gibt es eine Repräsentantenfunktion $c : V \to F$ mit $\forall v \in V : c(v) \in C(v)$.

Es sei $\{u, v\}$ die in Schritt 2a) gewählte Kante.
Wir betrachten jetzt sogenannte *Kantenfarbfolgen*, das sind mit u (bzgl. $\overline{E} \cup \{\{u, v\}\}$) inzidierende Knotenfolgen $x_0, \ldots, x_s$ mit $x_0 = v$ und $f(\{u, x_i\}) = c(x_{i-1})$. Eine Kantenfarbfolge heißt maximal, wenn es keine weitere Kanten $\{u, x\}$ in $\overline{E}$ gibt (mit $x \notin \{x_0, \ldots, x_s\}$) mit $f(\{u, x\}) = c(x_s)$.

Hilfssatz: Ist $x_0, \ldots, x_s$ eine Kantenfarbfolge mit $c(x_s) \in C(u)$, so ist f zu einer Färbung auf $\overline{E} \cup \{\{u, v\}\}$ erweiterbar.
Beweis: Setze $f(\{u, x_i\}) := c(x_i)$ für $0 \leq i \leq s$ und lasse sonst die alte Färbung bestehen. $\square$
Hinweis: Insbesondere für $s = 1$ ist das Färben trivial.

Die Situation vor (links) und nach (rechts) dem Umfärben mit dem Hilfssatz, $s = 4$:



In Polynomzeit kann nun eine maximale Kantenfarbfolge (an u) gefunden werden. Wegen des Hilfssatzes können wir $c(x_s) \notin C(u)$ annehmen.
Wegen der Maximalität der Folge gibt es ein $0 \leq i < s$ mit $f(\{u, x_i\}) = c(x_s)$, denn $c(x_s)$ ist ja nicht mehr „frei“ zum Färben von u-inzidenten Kanten.

Damit gilt: $c(x_{i-1}) = c(x_s)$.

p_{i-1}: ein maximaler Pfad in $(V, \overline{E})$, der bei x_{i-1} startet und dessen Kanten abwechselnd mit $c(u)$ und $c(x_s)$ gefärbt sind.
w: der letzte Knoten dieser Folge.
Unterscheide zwei Fälle:

(a) $w \neq u$. $\rightsquigarrow$ Färbe alle ursprünglich mit $c(u)$ gefärbten Kanten mit $c(x_s)$ und umgekehrt. Färbe Kante $\{u, x_{i-1}\}$ mit $c(u)$. Für die Kantenfarbfolge $x_0, \ldots, x_{i-1}$ ist der HS anwendbar.

(b) $w = u$. Jetzt suchen wir in Linearzeit einem weiteren maximalen Pfad p_s, der bei x_s startet und dessen Kanten abwechselnd mit $c(u)$ und $c(x_s)$ gefärbt sind, mit letztem Knoten w'.
Gilt $w \neq u$, können wir p_s analog zu (a) umfärben und dann den Hilfsatz anwenden.
Diskutiere Fall $w = w' = u$. Da p_{i-1} und p_s maximal und die Färbung insbesondere an u-Kanten zulässig war, müssen auch die vorletzten Knoten von p_{i-1} und p_w gleich sein. Das Argument wiederholt sich, bis o.E. x_s auf p_{i-1} erscheint. Da p_{i-1} nicht endet, gibt es eine mit $c(x_s)$ gefärbte Kante, die inzident zu x_s ist. $\rightsquigarrow$ Widerspruch zu $c(x_s) \in C(x_s)$. $\square$

Satz: Minimales Kantenfärben gehört zu PTAS$^\infty$.

Beweis: Da $\Delta(G) \leq m^*(G)$, gilt für die Lösung $m(G)$ unseres Algorithmus $m(G) \leq \Delta(G) + 1 \leq m^*(G) + 1$. Daher ist:

$$\frac{m(G)}{m^*(G)} \leq \frac{m^*(G)+1}{m^*(G)} = 1 + \frac{1}{m^*(G)}.$$

□

Mitteilung: Mit Hilfe der Gap-Technik lässt sich zeigen, dass das Kantenfärbeproblem kein PTAS besitzt, sofern nicht $P = NP$.

Bin Packing (BP)

Idee: polynomielle Lösbarkeit der folgenden eingeschränkten Variante:

Minimum (c, δ) *eingeschränktes BP* (für $c \in \mathbb{N}, c > 0$ und $\delta \in \mathbb{Q}$, $\delta \leq 1$) liegt vor, falls
es höchstens c unterschiedliche Größen der Gegenstände gibt und
jeder Gegenstand wenigstens δ groß ist (also einen Bruchteil der mit 1 angenommenen Kapazität jedes Behälters).

Um Bruchzahlen zu vermeiden bei Angabe der Gegenstandsgrößen s_i, gestatten wir noch die Angabe der Behältnisgröße B, weichen also formal von unserer Ursprungsdefinition von Bin-Packing leicht ab. In diesem Sinne wäre $(I = \{3 : 4, 5 : 2, 7 : 1\}, B = 8)$ eine Instanz von Minimum $(3, 3/8)$-eingeschränktem BP.

Lemma: Minimum (c, δ)-eingeschränktem BP kann in der Zeit $O(n^q)$ gelöst werden, wobei n die Zahl der Gegenstände in der Eingabeinstanz ist und q nur von c und δ, nicht aber von n abhängt.

Beweis: Es sei (I, B) eine Instanz von Min.(c, δ)-eingeschränktem BP. Der **Typ** eines Behälters ist ein c-dimensionaler Vektor $\vec{b} = (t_1, \ldots, t_c)$ von natürlichen Zahlen mit $0 \le t_i \le n_i$, sodass $\sum_{i=1}^{c} t_i s_i \le B$.
Für jeden Typ gilt wegen $\delta B \le s_i$:

$$\sum_{i=1}^{c} t_i \quad \le \quad \frac{1}{\delta B} \sum t_i s_i \le \frac{1}{\delta}$$

Erinnerung

$$\left| \left\{ (x_1, \ldots, x_s) \mid x_i \in \mathbb{N}, x_i \ge 0, \sum x_i \le m \right\} \right| = \binom{m+s}{m} = \binom{m+s}{s}$$

Anwendung: Es gibt höchstens $q = \binom{c + \lfloor \frac{1}{\delta} \rfloor}{c}$ viele Typen von Behältern. Eine zulässige Lösung von Minimum (c, δ)-eingeschränktem BP kann daher durch einen q-dimensionalen Vektor $\vec{y} = (y_1, \ldots, y_q)$ beschrieben werden, wobei $y_i \geq 0$ angibt, wie viele Behälter von Typ i in der Lösung vorkommen. Da trivialer Weise höchstens n Behälter verwendet werden müssen, gibt es daher höchstens n^q verschiedene zulässige Lösungen. In diesem Raum kann in der Zeit $O(n^q)$ nach einer optimalen Lösung gesucht werden. $\square$

Ein PTAS$^\infty$ für Bin-Packing (Skizze)

1. Entferne „kleine Gegenstände“.

2. Gruppiere die verbleibenden Gegenstände in eine konstante Zahl von Größenklassen.

3. Finde optimale Lösung für verbleibende Instanz. (wie eben)

4. Mache die Gruppierung aus Schritt 2 rückgängig.

5. Füge die „kleinen Gegenstände“ wieder ein.

Zum Gruppieren der Gegenstände

Es sei x eine Bin-Packing-Instanz.
Die Gegenstände $x_1, \dots, x_n$ seien der Größe nach absteigend sortiert. Für jede natürliche Zahl $k \leq n$ sei $m := \lfloor n/k \rfloor$.
Teile die n Gegenstände auf $m(+1)$ Gruppen G_i auf mit $G_i = \{x_{(i-1)k+1}, \dots, x_{ik}\}$, $1 \leq i \leq m$ und $G_{m+1} = \{x_{mk+1}, \dots, x_n\}$.
Wir definieren nun eine neue Instanz x_g^k von BP (mit derselben Behältergröße), sodass x_g^k für jeden Gegenstand x_j von der ursprünglichen Instanz, der aus G_i war, einen Gegenstand der Größe $s(x_{(i-1)k+1})$ enthält für $i = 2, \dots, m+1$.
Es gibt also höchstens mk Gegenstände in x_g^k, und diese Gegenstände haben höchstens m verschiedene Größen.
Wir ignorieren die k größten Gegenstände auf diese Weise.

Lemma: $m^*(x_g^k) \leq m^*(x) \leq m^*(x_g^k) + k$.

Beweis: Jede Lösung für x_g^k ist trivialer Weise eine für x, wenn wir weiter k Behälter für die ersten (größten) Gegenstände von x bereit halten.
Eine Lösung von x wiederum kann wie folgt in eine für x_g^k umgebildet werden:
(a) Entferne alle Gegenstände aus der letzten Gruppe,
(b) ersetze jeden Gegenstand aus G_i durch einen Gegenstand, der so groß ist wie x_{ik+1} (wenn nötig; G_{m+1} hat evtl. weniger Elemente als G_m). □

Zur Behandlung „kleiner Gegenstände“

Es sei x eine BP-Instanz. Für jedes $\delta \in \mathbb{Q}, \delta \in \left(0, \frac{1}{2}\right]$, sei x_δ die Instanz, die aus x durch Fortlassen aller Gegenstände, die kleiner als δB sind, entsteht. Haben wir nun eine Lösung für x_δ, die M Behälter benötigt, so benutzen wir FirstFit, um die kleinen Gegenstände wieder einzufügen.

Lemma: Auf diese Weise lässt sich in Polynomzeit eine Lösung für x finden, die höchstens $\max(M, (1 + 2\delta)m^*(x) + 1)$ viele Behälter benutzt.

Beweis: Wir unterscheiden zwei Fälle:

(a) Es wurden keine neuen Behälter durch FirstFit geöffnet $\to M$ Behälter reichen.

(b) Es wurden $M' \geq 1$ neue Behälter durch FirstFit geöffnet. Wie schon früher überlegt, enthalten alle Behälter (mit möglicher Ausnahme des zuletzt geöffneten) höchstens δB vielen freien Platz. Daher ist

$$(1-\delta) \underbrace{(M+M'-1)}_{\text{\# Behälter bis auf den letzten}} \leq \frac{\sum_{i=1}^{n} s(x_i)}{B} \leq m^*(x)$$

$$\rightsquigarrow M+M' \leq \frac{1}{1-\delta} m^*(x)+1 \quad \leq \quad (1+2\delta)m^*(x)+1$$

denn wegen $\delta \in \left(0, \frac{1}{2}\right]$ gilt $2\delta^2 \leq \delta$, also $1 \leq 1-2\delta^2+\delta$. □

Ein PTAS$^\infty$ für BP AsymptBP(x, B, r)

1. Falls $r \geq 2$ nimm NextFit (oder FirstFit).
2. $\delta := (r-1)/2$;
3. Sei x_δ die aus x durch Fortlassen von Gegenständen der Größe $< \delta B$ entstehende Instanz; n' sei die Zahl der Gegenstände in x_δ.
4. $k := \lceil (r-1)^2 n'/2 \rceil$;
5. Sei $x_{\delta,g}^k$ die aus x_δ durch Gruppierung gebildete Instanz.
6. Finde optimale Lösung $x_{\delta,g}^k$ vom Wert $m^*(x_{\delta,g}^k)$.
7. Füge die ersten (größten) k Gegenstände von x_δ in k „neue“ Behälter.
8. Wende FirstFit an, um die „kleineren Gegenstände“ wieder einzufügen.
9. Liefere die so erhaltene Packungsvorschrift zurück.

Satz: AsymptBP ist ein PTAS$^\infty$ für BP.

Beweis: AsymptBP läuft in Polynomzeit, da eine optimale Lösung für $x^k_{\delta,g}$ in Zeit n^q gefunden werden kann mit $q = f(\lceil n'/k \rceil, \delta)$, s. erstes Lemma.
Beachte: $\lceil n'/k \rceil$ hängt nicht von n ab, nur von r (bzw. δ).
Mit dem zweiten Lemma ist der Wert einer Lösung von AsymptBP beschränkt durch $m^*(x^k_{\delta,g}) + k$.
Alle Gegenstände in x_δ haben eine Größe von wenigstens δB, sodass $\delta n' \leq m^*(x_\delta)$ folgt, und damit

$$k \leq \frac{(r-1)^2}{2} n' + 1 = (r-1)\delta n' + 1 \leq (r-1) m^*(x_\delta) + 1.$$

Mit dem zweiten Lemma folgt $m^*(x^k_{\delta,g}) + k \leq m^*(x_\delta) + (r-1)m^*(x_\delta) + 1 = rm^*(x_\delta) + 1$.
Benutzen wir $r = (1 + 2\delta)$ im dritten Lemma, so erhalten wir, dass maximal

$$\max(rm^*(x_\delta) + 1, rm^*(x) + 1) \leq rm^*(x) + 1$$

viele Behälter von AsymptBP benutzt werden. □

Man kann sogar zeigen: BinPacking liegt in FPTAS$^\infty$.