Näherungsalgorithmen

(Approximationsalgorithmen)

WiSe 2006/07 in Trier

Henning Fernau
Universität Trier
fernau@informatik.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

FPTAS

Störend an den bisher in diesem Kapitel vorgestellten Approximationsschemata mag sein, dass die Laufzeit der Algorithmen exponentiell von der angestrebten Güte der Approximation abhängt —man möge dies überprüfen! Wir stellen jetzt einen „schöneren“ PTAS-Begriff vor.

Definition Es sei $\mathcal{P} \in \mathrm{NPO}$. Ein Algorithmus $\mathcal{A}$ heißt *volles Polynomzeit-Approximations-Schema* (engl. fully PTAS, kurz FPTAS), falls $\mathcal{A}$ für jedes Paar von Eingaben (x, r), x eine Instanz von $\mathcal{P}$ und $r > 1$, eine r-approximative Lösung zurück liefert in einer Zeit, die polynomiell in $|x|$ und in $\frac{1}{r-1}$ ist.
Die zugehörige Problemklasse nennen wir FPTAS.

Rucksack und FPTAS

Weiter oben haben wir bereits ein (einfaches) Beispiel für ein Problem aus FPTAS kennen gelernt, nämlich das Rucksackproblem.
Tatsächlich gibt es nur verhältnismäßig wenige (natürliche) Probleme in FPTAS; die meisten von ihnen sind Verwandte des Rucksackproblems.
Der Grund scheint darin zu liegen, dass man von einer ganzen Reihe von NPO-Problemen leicht nachweisen kann, dass sie **nicht** zu FPTAS gehören.

Polynomielle Beschränktheit und FPTAS

Ein Optimierungsproblem heißt *polynomiell beschränkt*, wenn es ein Polynom p gibt derart, dass (für jede Instanz $x \in I_{\mathcal{P}}$ und jede zulässige Lösung $y \in S_{\mathcal{P}}(x)$) $m(x, y) \leq p(|x|)$ gilt.

Satz: Gilt P $\neq$ NP, so gehört kein NP-hartes polynomiell beschränktes Optimierungsproblem zu FPTAS.

Beweis: Angenommen, es gibt ein FPTAS $\mathcal{A}$ für das Maximierungsproblem $\mathcal{P}$, dessen Laufzeit durch $q(|x|, \frac{1}{r-1})$ für ein Polynom q beschränkt ist (für jede Instanz x und jedes $r > 1$). Da $\mathcal{P}$ polynomiell beschränkt ist, gibt es ein Polynom p mit $m^*(x) \leq p(|x|)$[$*$] für jede Instanz x. Wir zeigen: für $r = 1 + \frac{1}{p(|x|)}$ liefert $\mathcal{A}(x, r)$ eine optimale Lösung für x (in Polynomzeit $q(|x|, p(|x|))$:

Da $m(x, \mathcal{A}(x, r))$ r-Approximation ist, gilt:

$$m(x, \mathcal{A}(x, r)) \geq m^*(x)\frac{p(|x|)}{p(|x|)+1} = m^*(x) - \frac{m^*(x)}{p(|x|)+1} \overset{[*]}{>} m^*(x) - 1$$

Da $m(x, \mathcal{A}(x, r)), m^*(x) \in \mathbb{N}$ und $m^*(x)$ größtmöglich, folgt

$$m^*(x) = m(x, \mathcal{A}(x, r)).$$

Aus der NP-Härte von $\mathcal{P}$ folgt nun die Behauptung. □

NP-Härte und Pseudo-Polynomialität

Eine gewisse Sonderrolle spielen Optimierungsprobleme, deren Schwierigkeit von der gewählten Codierung der (ganzen) Zahlen (unär oder binär) abhängt. Ist x eine Instanz eines NPO-Problems, so bezeichne $\max(\mathrm{x})$ den Betrag der betragsmäßig größten vorkommenden Zahl.

Definition: Ein NPO-Problem $\mathcal{P}$ heißt *pseudo-polynomiell*, wenn es eine Algorithmus $\mathcal{A}$ zu Lösung von $\mathcal{P}$ gibt, der, für jede Instanz x, mit einer Laufzeit arbeitet, die durch ein Polynom in $|\mathrm{x}|$ und $\max(\mathrm{x})$ beschränkt ist.

Beispiel: Das Rucksackproblem ist pseudo-polynomiell, wie oben vermerkt.
Für eine Instanz $x = (X = \{x_1, \ldots, x_n\}, \{p_1, \ldots, p_n\}, \{a_1, \ldots, a_n\}, b)$
würde $\max(x) = \max(p_1, \ldots, p_n, a_1, \ldots, a_n, b)$ sein.
Für die Laufzeit des früher betrachteten Algorithmus ist zu beachten:

$$O\left(n \sum_{i=1}^{n} p_i\right) \subseteq O\left(n^2 p_{\max}\right) \subseteq O\left(|x|^2 \max(x)\right).$$

Satz: Es sei $\mathcal{P} \in$ FPTAS für $\mathcal{P}$. Gibt es ein Polynom p, sodass für alle Instanzen $x \in I_{\mathcal{P}}$ gilt:

$$m^*(x) \leq p(|x|, \max(x)),$$

so ist $\mathcal{P}$ pseudo-polynomiell.

Beweis: Es sei $\mathcal{A}$ ein FPTAS für $\mathcal{P}$. Betrachte

$$\mathcal{A}'(x) = \mathcal{A}\left(x, 1 + \frac{1}{p(|x|, \max(x)) + 1}\right).$$

Aufgrund der Ganzzahligkeit der Lösungen liefert $\mathcal{A}'(x)$ ein Optimum. Ist die Laufzeit von $\mathcal{A}(x, r)$ durch $q\left(|x|, \frac{1}{r-1}\right)$ beschränkt, so ist die Laufzeit von $\mathcal{A}(x)$ durch $q(|x|, p(|x|, \max(x)) + 1)$ beschränkt, d.h. $\mathcal{P}$ ist pseudo-polynomiell. □

Definition: Es sei $\mathcal{P}$ ein NPO-Problem und p ein Polynom. $\mathcal{P}^{\max,\mathrm{p}}$ bezeichne die Einschränkung von $\mathcal{P}$ auf Instanzen x mit $\max(\mathrm{x}) \leq \mathrm{p}(|\mathrm{x}|)$. $\mathcal{P}$ heißt *stark NP-hart*, wenn es ein Polynom p gibt, sodass $\mathcal{P}^{\max,\mathrm{p}}$ NP-hart ist.

Satz: Ist P $\neq$ NP, so ist kein stark NP-hartes Problem pseudo-polynomiell.

Beweis: Angenommen, $\mathcal{P}$ ist ein stark NP-hartes Problem, das auch pseudo-polynomiell ist. Dann gibt es einen Algorithmus $\mathcal{A}$, der $\mathcal{P}$ in der Zeit $\mathrm{q}(|\mathrm{x}|, \max(\mathrm{x}))$ löst für ein geeignetes Polynom q und jede Instanz $\mathrm{x} \in \mathrm{I}_{\mathcal{P}}$. Für jedes Polynom p kann $\mathcal{P}^{\max,\mathrm{p}}$ daher in Zeit $\mathrm{q}(|\mathrm{x}|, \mathrm{p}(|\mathrm{x}|))$ gelöst werden. Da $\mathcal{P}$ stark NP-hart ist, müsste für ein p jedoch $\mathcal{P}^{\max,\mathrm{p}}$ NP-hart sein, woraus P $=$ NP folgt. $\square$.

Folgerung: Ist $\mathcal{P}$ ein stark NP-hartes NPO-Problem und gibt es ein Polynom p, sodass $\mathrm{m}^*(\mathrm{x}) \leq \mathrm{p}(|\mathrm{x}|, \max(\mathrm{x}))$ für jede Instanz $\mathrm{x} \in \mathrm{I}_{\mathcal{P}}$, so gilt $\mathcal{P} \notin$ FPTAS, wenn nicht P $=$ NP. $\square$

Die Sätze aus diesem Abschnitt gestatten es, für ein Problem zu schließen, dass es (unter gewissen komplexitätstheoretischen Annahmen) kein FPTAS besitzt. Dazu dient auch der folgende Zusammenhang, für dessen genaueres Verständnis wir auf die Vorlesung über parametrisierte Algorithmen verweisen.

Mitteilung: Besitzt das einem Optimierungsproblem $\mathcal{P}$ entsprechende parametrisierte Entscheidungsproblem *keinen* Festparameteralgorithmus, so liegt $\mathcal{P}$ nicht in FPTAS.

Zwischen APX und NPO

Wie oben gesehen, gibt es Probleme (wie das Handelsreisendenproblem), die sich (vorbehaltlich P $\neq$ NP) nicht bis auf einen konstanten Faktor mit einem Polynomzeitalgorithmus angenähert lösen lassen.
In solchen Fällen stellt sich die Frage, ob es denn wenigstens möglich ist, Leistungsgarantien für Approximationsalgorithmen zu finde, die von der Länge der Instanz abhängen.
In diesem Abschnitt lernen wir Beispiele für solche Näherungsalgorithmen kennen.
Betrachten wir zunächst eine weitere Verallgemeinerung des uns nun schon wohlbekannten Knotenüberdeckungsproblems:

Das Mengenüberdeckungsproblem (Set Cover, SC)

I : Eine Kollektion C von Teilmengen einer endlichen Grundmenge S.

S : Eine Mengenüberdeckung $C' \subseteq C$ für S, d.h. eine Teilkollektion, sodass jedes Element der Grundmenge S zu wenigstens einem Element der Teilkollektion gehört.

$\mathfrak{m}$: $|C'|$

opt : min

Zusammenhang mit Knotenüberdeckung

Das Knotenüberdeckungsproblem ist insofern ein Mengenüberdeckungsproblem, als dass dort die Grundmengenelemente „Kanten“ heißen und ein Element der Kollektion einem „Knoten“ entspricht, genauer der Menge von Kanten, die mit nämlichen Knoten inzidieren.

$\rightsquigarrow$ Der folgende Greedy-Algorithmus für das Mengenüberdeckungsproblem eine unmittelbare Verallgemeinerung eines Greedy-Algorithmus des Knotenüberdeckungsproblems, denn die Auswahl eines Elementes der Kollektion mit größter Mächtigkeit entspricht gerade der Wahl eines Knotens von größtem Grad.

In ähnlicher Weise lässt sich auch das Hitting-Set-Problem als Mengenüberdeckungsproblem auffassen (und umgekehrt).

GreedySC (S, C)

0. Teste, ob $\bigcup_{c \in C} c = S$ gilt.
1. $U := S$
2. Für jede Menge $c_i \in C$ setze $c'_i := c_i$;
3. $C' := \emptyset$;
4. Solange $U \neq \emptyset$ tue:
   - 4a. Wähle unter den c'_i eine Menge c'_j größter Mächtigkeit
   - 4b. $C' := C' \cup \left\{c^*_j\right\}$;
   - 4c. $U := U \setminus c'_j$;
   - 4d. Für jede Menge c'_i setze $c'_i := c'_i - c'_j$.
5. Liefere C' zurück.

Satz: Ist $n = |S|$, so ist GreedySC ein $(\lfloor \ln n \rfloor + 1)$-approximativer Polynomzeitalgorithmus für das Mengenüberdeckungsproblem.

Beweis: Starten wir zunächst mit einem kleinen mathematischen *Exkurs*:
$H(r) := \sum_{i=1}^{r} \frac{1}{i}$ bezeichnet die r*-te harmonische Zahl*.
Bekannt: $(H(r)) = \Theta(\log(r))$. Noch genauer gilt:

$$\lfloor \ln r \rfloor \leq H(r) \leq \lfloor \ln r \rfloor + 1$$

Um die Aussage des Satzes zu beweisen, werden wir zeigen:

$$\sum_{c_i \in C^*} H(|c_i|) \geq |C'| \qquad (*)$$

Dabei ist C^* irgendeine optimale Überdeckung und C' eine GreedyVC gelieferte Lösung. Wegen $|c_i| \leq n$ folgt aus $(*)$:

$$|C'| \leq \sum_{c_i \in C^*} H(|c_i|) \leq \sum_{c_i \in C^*} H(n) \leq |C^*| H(n) \leq |C^*| (\lfloor \ln n \rfloor + 1),$$

was die Satzbehauptung liefert. (Die Polynomzeitbehauptung ist trivial.)

Um $(*)$ nachweisen zu können, führen wir noch einige Bezeichnungen ein:

Ist $x = \left(S, \overbrace{\{c_1, \ldots, c_m\}}^{=C} \right)$ eine SC-Instanz, so sei $a_1, \ldots, a_{|C'|}$ die Folge der Indizes derjenigen Teilmengen aus der Kollektion, die zu $C' \subseteq C$ gehören, d.h.

$$C' = \left\{ c_{a_1}, \ldots c_{a_{|C'|}} \right\}.$$

Für jedes $j \in \{1, \ldots, |C'|\}, i \in \{1, \ldots, m\}$ sei c_i^j der „überlebende“ Teil von c_i, **bevor** die Teilmenge c_{a_j} gewählt worden ist durch GreedySC.
Damit ist $c_i^1 = c_i$ (für alle $i \in \{1, \ldots, m\}$).
Außerdem vereinbaren wir: $c_i^{|C'|+1} = \emptyset$ für alle $i \in \{1, \ldots, m\}$. Die Menge der Elemente von c_i, die das erste Mal durch c_{a_j} überdeckt werden, ist

$$c_i \cap c_{a_j}^j = c_i^j \cap c_{a_j}^j = c_i^j \setminus c_i^{j+1}. \qquad [+]$$

l_i sei der größte Index, für den $c_i^{l_i} \neq \emptyset$ gilt, d.h. $c_i^{l_i+1} = \emptyset$, m.a.W., nachdem $c_{a_{l_i}}$ zur Überdeckungs-Kollektion hinzugefügt wurde, sind alle Elemente von c_i abgedeckt.

Behauptung 1:

$$\forall i \in \{1, \ldots, m\} : H\left(|c_i|\right) \geq \sum_{j=1}^{|C'|} \frac{\left|c_i \cap c_{a_j}^j\right|}{\left|c_{a_j}^j\right|}$$

Behauptung 2: $\forall C'' \subseteq C, C''$ ist Mengenüberdeckung

$$\sum_{c_i \in C''} \sum_{j=1}^{|C'|} \frac{\left|c_i \cap c_{a_j}^j\right|}{\left|c_{a_j}^j\right|} \geq |C'|$$

Aus beiden Behauptungen folgt sofort $(*)$, denn Behauptung 2 gilt insbesondere für $C'' = C^*$, also ist

$$|C'| \leq \sum_{c_i \in C^*} \sum_{j=1}^{|C'|} \frac{\left|c_i \cap c_{a_j}^j\right|}{\left|c_{a_j}^j\right|} \leq \sum_{c_i \in C^*} H\left(|c_i|\right).$$

Beide Behauptungen ergeben sich nun nach einigen leichten algebraischen Manipulationen.

Zu Behauptung 1: Es sei $i \in \{1, \ldots, m\}$. Da GreedySC für jedes $1 \leq j \leq |C'|$ stets eine größte überlebende Menge wählt, gilt

$$[\$] \qquad \left|c_i^j\right| \leq \left|c_{a_j}^j\right|$$

in unserer Notation.
Also ist:

$$\sum_{j=1}^{|C'|} \frac{\left|c_i \cap c_{a_j}^j\right|}{\left|c_{a_j}^j\right|} \overset{[+]}{=} \sum_{j=1}^{C'} \frac{\left|c_i^j\right| - \left|c_i^{j+1}\right|}{\left|c_{a_j}^j\right|} \overset{[\$]}{\leq} \sum_{j=1}^{|C'|} \frac{\left|c_i^j\right| - \left|c_i^{j+1}\right|}{\left|c_i^j\right|} \overset{\text{Def. } l_i}{\leq} \sum_{j=1}^{l_i} \sum_{K=|c_i^{j+1}|+1}^{|c_i^j|} \frac{1}{|c_i^j|}$$

$$\overset{[\sharp]}{\leq} \sum_{j=1}^{l_i} \sum_{k=1}^{|c_i^j|-|c_i^{j+1}|} \frac{1}{k + |c_i^{j+1}|} \overset{|c_i^{j+1}|\geq 0}{\leq} \sum_{j=1}^{l_i} \sum_{k=1}^{|c_i^j|-|c_i^{j+1}|} \frac{1}{k} = \sum_{j=1}^{l_i} \left(H\left(\left|c_i^j\right|\right) - H\left(\left|c_i^{j+1}\right|\right)\right)$$

$$\overset{\text{Teleskop}}{=} H\left(\left|c_i^1\right|\right) - H\left(\left|c_i^{l_i+1}\right|\right) \overset{\text{s. o.}}{=} H\left(|c_i|\right)$$

Die Ungleichung $[\sharp]$ ist trivial, denn sie bedeutet:

$$\forall k = 1, \ldots, |c_i^j| - |c_i^{j+1}| : |c_i^j| \geq k + |c_i^{j+1}|.$$

Zu Behauptung 2:

$$\sum_{c_i \in C''} \sum_{j=1}^{|C'|} \frac{\left|c_i \cap c^i_{a_j}\right|}{\left|c^j_{a_j}\right|} = \sum_{j=1}^{|C'|} \frac{1}{\left|c^j_{a_j}\right|} \sum_{c_i \in C''} \left|c_i \cap c^j_{a_j}\right|$$

$$\overset{C'' \text{ Überdeckung!}}{\geq} \sum_{j=1}^{|C'|} \frac{1}{\left|c^j_{a_j}\right|} \cdot \left|c^j_{a_j}\right| = |C'|$$

Beim letzten $\geq$ ist $\neq$ möglich, falls gewisse Elemente mehrfach abgedeckt werden. □

Zwei Bemerkungen

— Die in obigem Satz angegebene Abschätzung der Leistungsgüte von GreedySC ist nicht verbesserbar. Es gibt eine Folge von Instanzen, für die die Schranke auch erreicht wird.
— GreedySC ist optimal in dem Sinne, dass es (unter einigen „wahrscheinlich“ komplexitätstheoretischen Annahmen) für kein $\epsilon > 0$ einen „$(\ln - \epsilon)$-Approximationsalgorithmus“ für SC gibt.

Graphenfärben

GreddyGC (G=(V,E))

1. $i := 0; U := V$
2. Solange $U \neq \emptyset$ tue:

   2a. $i := i + 1$;

   2b. $I :=$ GreedyIndependentSet$(G(U))$;

   2c. Färbe Knoten aus I mit „Farbe“ i;

   2d. $U := U \setminus I$
3. Liefere so erhaltene Färbung $f : V \to \mathbb{N}$ zurück.

Lemma: Ist $G = (V, E)$ k-färbbar, so benötigt GreedyGC höchstens $\frac{3|V|}{\log_k(|V|)}$ viele Farben zum färben von G.

Beweis: Es sei $H = G(U)$ irgendein im Schritt 2b. verarbeiteter Teilgraph von G. Da G k-färbbar ist, ist auch H k-färbbar. (wird fortgesetzt)

Hilfssatz: Wird ein Graph H GreedyIndependentSet als Eingabe gegeben, so wird eine unabhängige Menge $I \subseteq U$ geliefert, die wenigstens $\lceil \log_k (|U|) \rceil$ viele Knoten enthaÏt.

Beweis: Da H k-färbbar, enthält H eine unabhängige Menge $\bar{I}$ mit wenigstens $|U|/k$ Knoten. Jeder Knoten dieser Menge hat einen Maximalgrad $(|U| - |U|/k)$ (denn sonst gäbe es Verbindungen zwischen Knoten $\bar{I}$). Damit ist trivialerweise der Minimalgrad in H maximal $(|U| - |U|/k)$. Da als erster Knoten v_1 (für I) von GreedyIndependentSet einer minimalen Grades gewählt wird und er samt seinen höchstens $(|U| - |U|/k)$ vielen Nachbarn entfernt wird, wird der nächste Knoten v_2 in $H(U \setminus N[v_1])$ gesucht mit

$$|U \setminus N[v_1]| > |U| - (|U|/k + 1) = |U|/k - 1.$$

Da $H(U \setminus N[v_1])$ k-färbbar, lässt sich das Argument wiederholen.
Abgebrochen wird diese Suche, wenn U erschöpft ist.
Dazu sind mindestens $\lceil \log_k (|U|) \rceil$ viele Schritte nötig, und soviele Knoten werden auch wenigstens zu I hinzugenommen. □

Forsetzung des Beweises des Lemmas:

Hilfsatz $\rightsquigarrow$ wenigstens $\lceil \log_k(|U|) \rceil$ viele Knoten mit Farbe i gefärbt.
Wie groß ist U vor dem Durchlaufen der Schritte 2a.-2d.?

Fall 1: Solange $|U| \geq |V|/\log_k(|V|)$, gilt wegen $|V|/\log_k(|V|) > \sqrt{|V|}$:

$$\lceil \log_k |U| \rceil \geq \log_k |U| \geq \log_k (|V| / \log_k (|V|)) > \log_k \sqrt{|V|} = \frac{1}{2} \log_k (|V|)$$

Da U bei jedem Durchlauf der Schleife „2“ um wenigstens $\frac{1}{2} \log_k (|V|)$ abnimmt, kommt der Fall 1 höchstens $2|V| / \log_k (|V|)$ oft zur Anwendung, und dabei werden höchstens $2|V| / \log_k (|V|)$ viele Farben benutzt.

Fall 2: Sobald $|U| < |V|/\log_k(|V|)$, reichen trivialer Weise $|V|/\log_k(|V|)$ viele Farben aus.

Fall 1 und Fall 2 zusammen liefern die Behauptung. □

Satz: Ist $n = |V|$, so ist GreedyGC ein $O(n/\log(n))$-approximativer Algorithmus fürs Graphenfärben.

Beweis: Nach dem Lemma benötiget GreedyGC höchstens $3n/\log_k(n)$ viele Farben mit $k = m^*(G)$. Also ist

$$\frac{m(G, \text{GreedyGC}(G))}{m^*(G)} \leq \frac{3n \log(m^*(G)/\log(n)}{m^*(G)} \leq \frac{3n}{\log(n)}.$$

□

Bem.: Der soeben dargestellte $O(n/\log n)$-approximative Algorithmus ist nicht bestmöglich. So wurde ein $O\left(n\frac{(\log\log n)^2}{(\log n)^3}\right)$-approximativer Algorithmus gefunden. Auf der anderen Seite ist bekannt, dass es —sofern nicht P = NP— keinen $n^{1/7-\epsilon}$-approximativen Algorithmus geben kann für jedes $\epsilon > 0$. Insbesondere dürfte das Graphenfärbeproblem nicht in APX liegen.