Approximationsklassen für Optimierungsprobleme

Matthias Erbar

Wie genau ist ungefähr?
Sommerakademie Görlitz 2007

Gliederung

Motivation

- Viele Optimierungsprobleme sind nicht effizient exakt lösbar (NPO).
  Das macht Approximation notwendig.
- Wünschenswert ist eine Kontrolle der Approximationsgüte.
- Verschiedene Probleme zeigen unterschiedliche Approximierbarkeitseigenschaften:
  - Manche lassen sich beliebig gut effizient approximieren.
  - Viele jedoch gar nicht oder nur bis zu einer intrinsischen Grenze.

Ziel des Vortrags

- Ziel des Vortrags ist eine Klassifikation von NPO-Optimierungsproblemen nach ihrer Approximierbarkeit.
- Wir untersuchen Beispiele, die unterschiedliches Approximierungsverhalten illustrieren.
- $\mathcal{P} \neq \mathcal{NP}$ vorausgesetzt, erhalten wir eine strikte Hierarchie innerhalb von NPO.

Terminologie

Definition

Ein Optimierungsproblem P=(I,Sol,m) ist gegeben durch:

- I: Menge der Instanzen
- Sol: Funktion, die $x \in I$ die Menge der Lösungen von x zuordnet
- m: Maßfunktion, gibt Wert einer Lösung an

Der Optimalwert wird mit $m^*(x)$ bezeichnet.

Beispiel: Maximum Knapsack

- I: {Menge von Gegenständen x_i mit Wert p_i u. Größe g_i, Kapazität B}
- Sol(x): {Teilmenge von x mit $\sum g_i < B$}
- m: $\sum p_i$

absolute Fehlergrenzen

Definition

Algorithmus A heißt absoluter Approximationsalgorithmus zu Problem P, falls er für jedes $x \in I$ eine Lösung $A(x) \in Sol(x)$ liefert und Konstante k existiert, mit:

$$\forall x \in I: \ |m^*(x) - m(x, A(x))| < k \, .$$

- Meist zu starke Forderung. Zum Beispiel Maximum Knapsack erlaubt keinen solchen effizienten Algorithmus wie gestern gesehen.
- Also besser relativen Fehler betrachten…

relative Fehlergrenzen

Definition

Sei y Lösung zur Instanz x. performance ratio (PR) geg. durch:

$$r(x,y) = max(\frac{m^*(x)}{m(x,y)}, \frac{m(x,y)}{m^*(x)})$$

Definition

Ein Algorithmus A heißt r-Approx.algorithmus zu Problem P, falls:

$$\forall x \in I : \quad r(x, A(x)) < r \,.$$

Die Klasse APX

- Die Klasse aller NPO-Probleme, die einen polynomiellen r-Approx.Algorithmus zulassen, bezeichnen wir mit APX.
- APX ist strikte Unterklasse von NPO, falls $\mathcal{P} \neq \mathcal{NP}$, denn:

Theorem

Falls Minimum TSP zu APX gehört, so $\mathcal{P} = \mathcal{NP}$.

Die Klasse APX

Beweis.

Idee: Reduktion von Hamilton-Pfad ($\mathcal{NP}$-vollständig) auf TSP

- $G = (V, E)$ ger. Graph mit n Knoten, $r > 1$
- Erhalten Instanz von TSP mit G und Entfernungen:

$$d(v, w) = \begin{cases} 1 & (v, w) \in E \\ 1 + n * r & \text{sonst} \end{cases}$$

- Falls Hamilton-Pfad ex., so optimale Strecke n; nächste Approximative Lösung mind. $n(1 + r)$. Falls kein Hamilton-Pfad ex., so optimale Strecke mind. $n(1 + r)$.
- Da $PR < r$ liefert der Approximationsalgorithmus genau dann Lösung der Länge n, falls Hamilton-Pfad existiert.

□

Innerhalb APX zeigt sich unterschiedliches Approximationsverhalten:

- viele Probleme können nur bis zu einer inherenten Grenze effizient approximiert werden.
- Beweis solcher Resultate erfolgt oft analog zu $TSP \notin APX$. Werden die selbe Technik bei Minimum Bin Packing anwenden.
- manche Probleme erlauben effiziente Algorithmen mit beliebig guter PR. $\rightarrow$ Approximationschemata

polynomielle Approximationsschemata

Oft benötigt man variable Approximationsgüte. Für bessere PR wird höherer Rechenaufwand in Kauf genommen.

Definition

Ein Algorithmus A heißt polynomielles Approximationschema für ein Problem P, falls er für alle $x \in I, r > 1$ eine Lösung A(x,r) liefert mit $r(x, A(x, r)) < r$, wobei Laufzeit polynomiell in $|x|$.

Konstruktionsprinzip eines Approximationsschemas

- benutze einen r-approx.-Algorithmus
- löse eine ausreichend große Teilinstanz exakt
- wende den bekannten Alg. auf die Restinstanz an

Beispiel eines PTAS

Optimierungsproblem Partition

Menge X von Elementen x mit Gewichten w(x), soll in zwei möglichst gleich schwere Mengen aufgeteilt werden.

Partition PTAS

```
input: Menge X von Elementen mit ganzen Gewichten w_i, r > 1;
output: Partition von X in Y_1 u. Y_2;
begin
  if r > 2 then return X, leer
  else
    Sortiere Elemente in nicht aufsteigender Reihenfolge bezgl. Gewicht;
    k(r) := ⌈(2-r)/(r-1)⌉;
    Finde optimale Lösung Y_1, Y_2 für x_1, ..., x_k(r);
    for j:=k(r)+1 to n do
      if ∑_{Y_1} w_i < ∑_{Y_2} w_i then
        Y_1 := Y_1 ∪ {x_j}
      else
        Y_2 := Y_2 ∪ {x_j};
    return Y_1, Y_2;
  end;

end.
```

Beispiel eines PTAS

Analyse

- Algorithmus löst einen Teil optimal, nutzt Greedy-Algorithmus für den Rest.
- Polynomiell, denn k hängt nicht von $|x|$ ab.
- OBdA sei $r < 2$, $w(Y_1) \geq W(Y_2)$ und x_l das letzte zu Y_1 hinzugefügte Element. Setzte $L = \frac{w(X)}{2}$.
  1. Falls x_l in erster Phase hinzugefügt, so Lösung optimal
  2. Falls nicht, so $w(Y_1) \leq L + \frac{a_l}{2}$. Weiter gilt $w(X) \geq a_l(k(r)+1)$, denn: $a_l \leq a_j,\ 1 \leq j \leq k(r)$.
- Also: $\frac{w(Y_1)}{m^*(X)} \leq \frac{w(Y_1)}{L} \leq 1 + \frac{a_l}{2L} \leq 1 + \frac{1}{\frac{2-r}{r-1}+1} = r$

Die Klasse PTAS

- Die Klasse aller NPO-Probleme, die ein polynom. Approximationschema zulassen, bezeichnen wir mit PTAS.
- PTAS ist strikte Unterklasse von APX, falls $\mathcal{P} \neq \mathcal{NP}$, denn:

Theorem

Falls Minimum Bin Packing zu PTAS gehört, so $\mathcal{P} = \mathcal{NP}$.

Optimierungsproblem Bin Packing

Endliche Menge von Elementen u mit Größe g(u) soll passend in möglichst wenige Dosen mit Kapazität B gepackt werden.

Die Klasse PTAS

Beweis.

Idee: Reduktion von Partition ($\mathcal{NP}$-vollständig) auf Bin Packing mit Hilfe eines $(\frac{3}{2} - \epsilon)$-Approx.algorithmus

- Sei x Instanz von Partition, B Summe der Gewichte.
- Erhalten Instanz x' von Bin Packing durch: für jedes Element von x mit Gewicht w hat x' Element der Größe $\frac{2w}{B}$
- Falls x partitionierbar, so $m^*(x') = 2$, sonst $m^*(x') = 3$
- r-Approximationsalgorithmus mit $r < \frac{3}{2}$ liefert genau dann Lösung mit 2 Bins, falls x partitionierbar.

$\square$

Die Klasse FPTAS

- Wächst die Laufzeit sehr schnell mit der PR-Schranke, so ist ein PTAS nicht sehr nützlich.
- Wünschenswert ist polynom. Abhängigkeit von $|x|$ und $\frac{1}{r-1}$.

Definition

Ein Algorithmus A heißt vollständig polynomielles Approximationschema für ein Problem P, falls er für alle $x \in I, r > 1$ eine Lösung A(x,r) liefert mit r(x,A(x,r))<r, wobei Laufzeit polynomiell in $|x|$ und $\frac{1}{r-1}$.

- Die Klasse aller NPO-Probleme, die ein vollständig polyn. Approx.schema zulassen, bezeichnen wir mit FPTAS.
- Wieder gilt: $FPTAS \subsetneq PTAS$, falls $\mathcal{P} \neq \mathcal{NP}$.

erweiterte Klassen

Schranken an die performance ratio lassen sich aufweiten:

- Statt konstanter Schranken (APX) kann man langsam wachsende Funktionen der Inputgröße als Schranken betrachten. Zum Bespiel $\mathcal{O}(\frac{n}{log(n)})$.
- Statt PTAS kann man Schemata betrachten, deren PR asymptotisch für große Werte optimaler Lösungen beschränkt sind.

Die Klasse $PTAS^{\infty}$

Definition

Ein Algorithmus A heißt asymptotisches polynomielles Approximationschema für ein Problem P, falls eine Konstante k existiert, so daß

$$\forall x \in I, r > 1 : \; r(x, A(x, r)) \leq r + \frac{k}{m^{*}(x)} ,$$

wobei Laufzeit polynomiell in $|x|$.

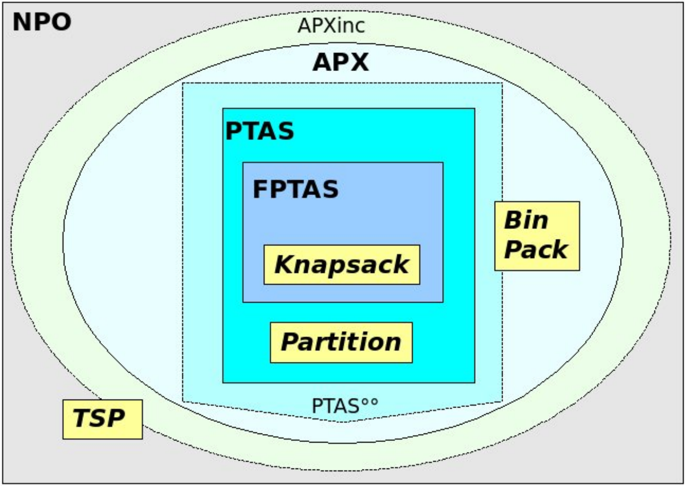
Zum Beispiel ist Minimum Bin Packing in $PTAS^{\infty}$.

Zusammenfassung

NPO-Probleme lassen sich nach ihrer Approximierbarkeit in Klassen einteilen:

- APX: in polynomieller Zeit approximierbar mit garantiertem rel. Fehler
- PTAS: bei beliebig vorgeg. rel. Fehler in polyn. Zeit approximierbar
- FTPAS: wie PTAS, Laufzeit hängt polyn. von Genauigkeit ab
- Diese Klassen bilden eine strikte Hierarchie, falls $\mathcal{P} \neq \mathcal{NP}$.

Übersicht


NPO
APXinc
APX
PTAS
FPTAS
Knapsack
Partition
Bin Pack
PTAS°°
TSP

Literatur I

G. Ausiello
Complexity and Approximation.
Springer, 1999.