Approximationsalgorithmen
SoSe 2019

Benedikt Lüken-Winkels

April 23, 2019

Contents

1 1.Vorlesung

Foliensatz 1

1.1 Orga

- **Sprechstunde** Do, 13-14 Uhr
- **Vorlesung** Di, 12:15-13:45
- **Übung** Di, 8:15-9:45 Uhr (erster Termin 16.04.)
- **Prüfung** Mündl Prüfung

1.2 Einführung

1.2.1 Motivation

- wenn P $\neq$ NP, kan man keinen guten oder schnellen Algorithmus schreiben
- Zeigt man, dass ein Problem NP-schwer ist, kann kein schneller Algorithmus geschrieben werden

$\Rightarrow$ Heuristische Verfahren (keine mathematische Garantie). Warum funktionieren die Heuristiken so gut? Herangehensweisen

- Greedy Verfahren
- Randomisierte Verfahren: finden der Lösung mit hoher Wahrscheinlichkeit
- Parametrisierte Verfahren: exakte Lösungen und Versuch, den exponentiellen Teil gering zu halten
- Näherungsverfahren: Heuristiken mit Leistungsgarantie

Klasse von Problemen die zur Betrachtung stehen.
Quatrupel (I_P, S_P, m_P, opt_P) zur Beschreibung eines Optimierungsproblems

- I_P: geeignete Instanz eines Problems, genauer: "geeignet binär-codierte formale Sprachen".
- S_P: Bildet auf Menge der möglichen Lösungen ab
- m_P: x Instanz und y eine Lösung. Abbildung auf Maßzahl
- opt_P: Möglichst kleines Ergebnis oder möglichst großes
- $S_P^* : I_P \rightarrow$ Menge der bestmöglichen Lösungen

- m_P^* Wert oder Grenzwert einer bestmöglichen Lösung
- * bedeutet idR bestmöglich

⇒ **Ziel**: Leistungsgröße (Folie 15) ist 1, wenn Lösung optimal ist

1.2.2 Beispiel: Knotenüberdeckung

Möglichst wenige Knoten, um alle Kanten abzudecken

- Zuordnung zu den Optimierungsparametern Folie 17
- Verschiedene Beobachtungen zur Optimierung
  - Zwei Knoten im Dreieck gehören dazu
  - Bei Knoten mit Grad 1 wird immer der Nachbar genommen
- Auswählen eines Knotens bedeutet, dass diese Teile abgeschnitten werden
- ⇒ Vereinfachung des Graphen, zB neue Grad 1 Knoten

Greedyverfahren, GreedyVC (Folie 23)

- Änderung der Grade bei Durchführung
- Problem: Implementierung der Kantenlöschung (Kopieren des Graphen bei jeder Iteration nötig?)
- Folie 24: Lösung insofern (inklusions-) minimal, als dass das Entfernen eines Knotens keine andere Lösung zulässt

Suchbaumverfahren, Entscheidungsproblem (Folie 25) Liefert exakte Lösungen

- Zusätzlicher Parameter k ("Budget")
- Zwei Abbruchskriterien:
  - Alle Kanten abgedeckt
  - Nicht alle Kanten abgedeckt, aber $k = 0$
- Suchbaum im worst-case ein vollständiger Binärbaum, **aber** höchsten 2^k Schritte im Baum, da die Tiefe durch k begrenzt ist

Näherungsverfahren (Folie 30) Suchbaumverfahren ohne Fallunterscheidung. (Faktor 2-Approximations-Verfahren)

- Bei jeder Kante muss einer der Knoten in die Überdeckung
- Lokaler Fehler höchsten Faktor 2
- Zufall bei der Auswahl der Kanten kann zum Vorteil sein

Näherung gibt Schranke für die minimale Lösung dadurch, dass Heuristik eine Faktor 2 Lösung zeigt. $\Rightarrow$ (Folie 31) Lösung mit 22 Knoten zeigt eine optimale Lösung mit 11 Knoten

1.2.3 Beispiel: MAXSAT (Folie 32)

$mP =$ Anzahl der Klauseln, die die Formel erfüllen

Einfacher Ansatz

- Alles 0 und alles 1 setzen, dann das bessere Ergebnis zurückliefern
- $\Rightarrow$ liefert 2-Approximation

1.2.4 Beispiel: Unabhängige Knotenmengen (Folie 34)

Sehr schwer approximierbar

1.2.5 Beispiel: Unabhängige Kantenmengen (Folie 35)

Lösung in Polinomialzeit, um eine untere Schranke für die Knotenüberdeckung zu finden

2 2. Vorlesung

2.Foliensatz

2.1 Definition Gewichtsreduktionsfunktion

Eine Reduktion verringert die Gewichtsfunktion: $\forall x \in X : 0 \leq \delta(x) \leq w(x)$
Eine Reduktion ist **r-effektiv**, wenn $\delta(X) \leq r \cdot OPT(\delta)$

2.2 Allgemeines (gewichtetes) Überdeckungsproblem

- Grundmenge X
- Monotone Abbildung (Bewerung: 1 = Überdeckung oder 0) $f : 2^X \rightarrow \{0, 1\}$
- Gewichtsfunktion $w \rightarrow \mathbb{R}^+$ weist den Knoten ein Gewicht zu

- ⇒ Überdeckung mit kleinstmöglichem Gewicht
- Gewichtsreduktionsfunktion δ
- $OPT(w) = w(C^*)$ C^* ist optimale Überdeckung

Einfachere Problemanalyse durch Zerlegung von Gewichtsfunktionen in Untergewichtsfunktionen

2.3 Reduktion Bar-Yehuda, Even Folie 13

2-Approximation, Reduktion für jede Kante $\delta_e(v)$ wird angewandt auf jeden anliegenden Knoten

- Wähle das Minimum der Knoten als Gewicht für die Kante
- Nehme eine Kante und ziehe das Gewicht der Kante von den Knoten ab ⇒ einer der Knoten hat Grad 0 und damit Teil einer Überdeckung
- Nächster Schritt $w - \delta_e$, bedeutet, dass die Gewichtsfunktion verändert wird und eine neue Iteration beginnt

2.4 Reduktion Clarkson Folie 22

2-Approximation, Gewichtsreduktion über Knoten

- $\varepsilon(v) = \frac{w(v)}{d(v)}$
- Anliegende Knoten von v erhalten Gewicht $\varepsilon(v)$
- ⇒ $w - \delta_v$

2.5 Randomiesierte Verfahren

2-Approximation, Gewichtsreduktion über Knoten

- Zufallsalgorithmus gemäß r-effektiver Verteilung (nicht immer Faktor r, aber im Mittel erreicht)
- Implementierung der Intuition, dass großgradige Knoten interessant sind
- Bei ungewichteten Graphen:
  - $(w(v) = 1)$
  - Wahrscheinlichkeit einen Knoten zu wählen, $\frac{d(v)}{2|V|}$ ($2|V|$, weil alle Kanten Doppelt abgezählt werden)
  - Knoten mit großem Grad werden häufig, aber **nicht immer** in die Überdeckung aufgenommen

2.6 $\triangle$-Hitting-Set

Δ = maximaler Grad der Kanten (Wieviele Knoten hängen an einer Kante). $\Delta = 2$ quasi Knotenüberdeckungsproblem

Sonderfälle

- leere Kante (keine Knoten) $\Rightarrow$ keine Überdeckung möglich
- Kante mit nur einem Knoten $\Rightarrow$ automatisch hinzufügen

2.6.1 Beispiel "Smart Home"

System

- Systembestandteile C
- Systembeschreibung SD (wie das System sein sollte)
- beobachtetes Systemverhalten OBS

Ist ein Widerspruch in der Annahme, dass das System fehlerfrei funktioniert, finde die größte Menge an Systembestandteilen, sodass der Widerspruch verschwindet. (Schneide den defekten Teil des Systems ab)

2.6.2 Datenreduktion

- Kante f ist echte Teilmenge von Kante e $\Rightarrow$ entferne e
- Kante e ist gleich Knoten v $\Rightarrow$ Knoten ist in der Überdeckung
- Konten x hat ist nur in einer Kante mit Knoten y $\Rightarrow$ entferne x

3 3. Vorlesung

3.1 Problemvarianten

P_C **Konstruktionsproblem** Zur Instanz eines Problems soll eine bestmögliche Lösung und deren Wert geliefert werden. Häufig existieren mehrere beste Lösungen mit gleichem Wert.

P_E **Auswertungsproblem** Zur Instanz eines Problems wird nur der Wert der bestmögl Lösung benötigt.

P_D **Entscheidungsproblem** Ist der Wert einer optimalen Lösung größer k oder kleiner.

3.2 Klassen

NPO

- Finden der Lösung, Entscheidung, ob die Lösung gültig ist und Maßzahl/Messfunktion sind in Polinomialzeit lösbar.
- Größe der Lösung ist durch die Größe der Instanz gedeckelt
- Lösen des NP Problems in nicht deterministischer Weise (Raten)

Turing-Reduzierbarkeit Aufteilung des Problems in P_1 und P_2. P_2 ist ein Unterprogramm, das bei der Lösung einer Probleminstanz von P_1 hilft. Bei einem NPO-Problem: $P_D =^P P_E \leq^P P_C$. Ist P NP-Hart, gilt $P_C \leq^P P_D$

MAXCLIQUE (Lösung als Konstruktionsproblem mit Hilfe des Auswertungsproblems) Durch den Wert des Auswertungsproblems (P_E) kann die größte Clique gefunden werden. Die **Orakelfunktion** liefert die Größe der Lösung.

MinSAT

- Boolsche Formel als Instanz
- Maß ergibt sich aus all den Klauseln, die true ergeben
- Erfüllte Klauseln möglich gering halten.

Lösung Vertex Cover kann in MinSAT überführt werden und umgekehrt. Lösung für das eine Problem ist eine Lösung für das Andere Porblem