Vorlesung

Theoretische Informatik II

Bernhard Beckert

Institut für Informatik

UNIVERSITÄT KOBLENZ · LANDAU

Wintersemester 2007/2008

Dank

Diese Vorlesungsmaterialien basieren zum Teil auf den Folien zu den Vorlesungen von

Katrin Erk (gehalten an der Universität Koblenz-Landau)

Jürgen Dix (gehalten an der TU Clausthal)

Christoph Kreitz (gehalten an der Universität Potsdam)

Ihnen gilt mein herzlicher Dank.

– Bernhard Beckert, Oktober 2007

Teil VI

Teil VI

Motivation

Viele relevante Probleme nachweislich schwierig

Unentscheidbar Terminierung, Korrektheit von Programmen
Allgemeingültigkeit prädikatenlogischer Formeln

PSPACE-vollständig Spiele, Marktanalysen

NP-vollständig SAT, Navigation, Scheduling

Motivation

Viele relevante Probleme nachweislich schwierig

Unentscheidbar Terminierung, Korrektheit von Programmen
Allgemeingültigkeit prädikatenlogischer Formeln

PSPACE-vollständig Spiele, Marktanalysen

NP-vollständig SAT, Navigation, Scheduling

Motivation

Viele relevante Probleme nachweislich schwierig

Unentscheidbar Terminierung, Korrektheit von Programmen
Allgemeingültigkeit prädikatenlogischer Formeln

PSPACE-vollständig Spiele, Marktanalysen

NP-vollständig SAT, Navigation, Scheduling

Motivation

Lösungsansätze

Heuristische Lösung Nutze zusätzliches Wissen über die Struktur der in einer bestimmten Anwendung auftretenden Probleminstanzen; Verzichte auf Lösung im Allgemeinen zugunsten eines guten Average Case in der bestimmten Anwendung

Approximation Bestimme Näherungslösung
Verzichte auf optimale Antwort zugunsten kürzerer Laufzeit

Probabilistisch Bestimme richtige Lösung mit bestimmer (hoher) Wahrscheinlichkeit
Verzichte auf sichere Korrektheit der Antwort zugunsten kürzerer Laufzeit

Motivation

Lösungsansätze

Heuristische Lösung Nutze zusätzliches Wissen über die Struktur der in einer bestimmten Anwendung auftretenden Probleminstanzen; Verzichte auf Lösung im Allgemeinen zugunsten eines guten Average Case in der bestimmten Anwendung

Approximation Bestimme Näherungslösung
Verzichte auf optimale Antwort zugunsten kürzerer Laufzeit

Probabilistisch Bestimme richtige Lösung mit bestimmer (hoher) Wahrscheinlichkeit
Verzichte auf sichere Korrektheit der Antwort zugunsten kürzerer Laufzeit

Motivation

Lösungsansätze

Heuristische Lösung Nutze zusätzliches Wissen über die Struktur der in einer bestimmten Anwendung auftretenden Probleminstanzen;
Verzichte auf Lösung im Allgemeinen zugunsten eines guten Average Case in der bestimmten Anwendung

Approximation Bestimme Näherungslösung
Verzichte auf optimale Antwort zugunsten kürzerer Laufzeit

Probabilistisch Bestimme richtige Lösung mit bestimmer (hoher) Wahrscheinlichkeit
Verzichte auf sichere Korrektheit der Antwort zugunsten kürzerer Laufzeit

Approximation

Viele NP-harte Probleme haben Optimierungsvariante

Knapsack Finde möglichst kleines Gewicht für festen Minimalnutzen

CLIQUE Finde möglichst große Clique in einem Graphen

Nicht alle NP-harten Probleme lassen sich gut approximativ mit polynomiellem Aufwand lösen

Knapsack Es ist möglich, eine Näherungslösung in

$$O(n^3 * \varepsilon^{-1})$$

zu finden, wobei ε der relative Fehler ist

CLIQUE Keine gute polynomielle Approximation möglich (es sei denn **P** = **NP**)

Approximation

Viele NP-harte Probleme haben Optimierungsvariante

Knapsack Finde möglichst kleines Gewicht für festen Minimalnutzen

CLIQUE Finde möglichst große Clique in einem Graphen

Nicht alle NP-harten Probleme lassen sich gut approximativ mit polynomiellem Aufwand lösen

Knapsack Es ist möglich, eine Näherungslösung in

$$O(n^3 * \varepsilon^{-1})$$

zu finden, wobei ε der relative Fehler ist

CLIQUE Keine gute polynomielle Approximation möglich (es sei denn $\mathbf{P} = \mathbf{NP}$)

Approximation

Viele NP-harte Probleme haben Optimierungsvariante

Knapsack Finde möglichst kleines Gewicht für festen Minimalnutzen

CLIQUE Finde möglichst große Clique in einem Graphen

Nicht alle NP-harten Probleme lassen sich gut approximativ mit polynomiellem Aufwand lösen

Knapsack Es ist möglich, eine Näherungslösung in

$$O(n^3 * \varepsilon^{-1})$$

zu finden, wobei ε der relative Fehler ist

CLIQUE Keine gute polynomielle Approximation möglich (es sei denn **P** $=$ **NP**)

Probabilistische Algorithmen

Idee

- Indeterministische, zufallsgesteuerte Rechnung
- Ziel dabei: Falsche Entscheidung möglich aber unwahrscheinlich
- Verringerung der Fehlerwahrscheinlichkeit durch Wiederholung der Rechnung
- Merke: 2^{-100} liegt schon unter der Wahrscheinlichkeit von Hardwarefehlern

Probabilistische Algorithmen

Idee

- Indeterministische, zufallsgesteuerte Rechnung
- Ziel dabei: Falsche Entscheidung möglich aber unwahrscheinlich
- Verringerung der Fehlerwahrscheinlichkeit durch Wiederholung der Rechnung
- Merke: 2^{-100} liegt schon unter der Wahrscheinlichkeit von Hardwarefehlern

Probabilistische Algorithmen

Idee

- Indeterministische, zufallsgesteuerte Rechnung
- Ziel dabei: Falsche Entscheidung möglich aber unwahrscheinlich
- Verringerung der Fehlerwahrscheinlichkeit durch Wiederholung der Rechnung
- Merke: 2^{-100} liegt schon unter der Wahrscheinlichkeit von Hardwarefehlern

Probabilistische Algorithmen

Idee

- Indeterministische, zufallsgesteuerte Rechnung
- Ziel dabei: Falsche Entscheidung möglich aber unwahrscheinlich
- Verringerung der Fehlerwahrscheinlichkeit durch Wiederholung der Rechnung
- Merke: 2^{-100} liegt schon unter der Wahrscheinlichkeit von Hardwarefehlern

Beispiel: Probabilistischer Algorithmus für 3-CLIQUE

NB: 3-CLIQUE ist polynomiell lösbar (anders als allgemein CLIQUE)

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

Wiederhole folgendes k mal:

- Wähle zufällig $v_1 \in V$ und $\{v_2, v_3\} \in E$
- Teste, ob v_1, v_2, v_3 eine 3-CLIQUE bilden

Fehlerwahrscheinlichkeit

$k = (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 0,5$

$k = 100 \cdot (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 2^{-100}$

Probabilistische Algorithmen

Beispiel: Probabilistischer Algorithmus für 3-CLIQUE

NB: 3-CLIQUE ist polynomiell lösbar (anders als allgemein CLIQUE)

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

Wiederhole folgendes k mal:

- Wähle zufällig $v_1 \in V$ und $\{v_2, v_3\} \in E$
- Teste, ob v_1, v_2, v_3 eine 3-CLIQUE bilden

Fehlerwahrscheinlichkeit

$k = (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 0,5$

$k = 100 \cdot (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 2^{-100}$

Probabilistische Algorithmen

Beispiel: Probabilistischer Algorithmus für 3-CLIQUE

NB: 3-CLIQUE ist polynomiell lösbar (anders als allgemein CLIQUE)

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

Wiederhole folgendes k mal:

- Wähle zufällig $v_1 \in V$ und $\{v_2, v_3\} \in E$
- Teste, ob v_1, v_2, v_3 eine 3-CLIQUE bilden

Fehlerwahrscheinlichkeit

$k = (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 0,5$

$k = 100 \cdot (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 2^{-100}$

Probabilistische Algorithmen

Beispiel: Probabilistischer Algorithmus für 3-CLIQUE

NB: 3-CLIQUE ist polynomiell lösbar (anders als allgemein CLIQUE)

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

Wiederhole folgendes k mal:

- Wähle zufällig $v_1 \in V$ und $\{v_2, v_3\} \in E$
- Teste, ob v_1, v_2, v_3 eine 3-CLIQUE bilden

Fehlerwahrscheinlichkeit

$k = (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 0,5$

$k = 100 \cdot (|E| \cdot |V|)/3$: Fehlerwahrscheinlichkeit $< 2^{-100}$

Probabilistische Algorithmen

Beispiel: Probabilistischer Algorithmus für 3-CLIQUE

NB: 3-CLIQUE ist polynomiell lösbar (anders als allgemein CLIQUE)

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

Wiederhole folgendes k mal:

- Wähle zufällig $v_1 \in V$ und $\{v_2, v_3\} \in E$
- Teste, ob v_1, v_2, v_3 eine 3-CLIQUE bilden

Fehlerwahrscheinlichkeit

$k = (|E| \cdot |V|)/3$**:** Fehlerwahrscheinlichkeit $< 0,5$

$k = 100 \cdot (|E| \cdot |V|)/3$**:** Fehlerwahrscheinlichkeit $< 2^{-100}$