Informatik III - Tutorium XIII & XV (SR -107)

Tut Nr. 9 – Üb8, Approximationsalgorithmen

David Münch

Universität Karlsruhe (TH)
Institut für Informatik
ITI Wagner

9. Januar 2008

Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

Organisatorisches

Email: muenchdavid@gmail.com

https://www.stud.uni-karlsruhe.de/~uhbro/

Tutorium 13: Mittwochs 8:00 Uhr - Raum -107
Tutorium 15: Mittwochs 9:45 Uhr - Raum -107

Übungsblattabgabe Donnerstag.

Was wollen wir heute erreichen?

Was wollen wir heute erreichen?

- Nichtexistenzbeweise von Approximationsalgorithmen mit absoluter Gütegarantie.

Was wollen wir heute erreichen?

- Nichtexistenzbeweise von Approximationsalgorithmen mit absoluter Gütegarantie.
- Beweis der relativen Gütegarantie bei Approximationsalgorithmen mit rel. Gütegarantie.

Aufgabe 6

Zeigen Sie:

a) Aus $\mathcal{P} = \mathcal{NP}$ folgt $\mathcal{P} \setminus \{\emptyset, \Sigma^*\} = \mathcal{NPC}$.

b) Warum sind $\emptyset$ und Σ^* (insbesondere unter dieser Annahme) nicht $\mathcal{NP}$-vollständig?

Aufgabe 6

a) Seien:
$L \in \mathcal{P} \backslash \{\emptyset, \Sigma^*\}$ Sprache über Σ und
$L' \in \mathcal{NP}$ Sprache über Σ'.
Wir zeigen nun: $L' \propto L$, falls $\mathcal{P} = \mathcal{NP}$:
Weil $L' \in \mathcal{NP} = \mathcal{P}$ existiert eine polynomial beschränkte DTM M' für L'. Somit können wir eine polynomiale Transformation $\tau : \Sigma'^* \to \Sigma^*$ der Sprache L' in die Sprache L angeben. Da aber L weder $\emptyset$ noch Σ^* ist, gibt es Wörter $w_J, w_N \in \Sigma^*$ mit $w_J \in L$ und $w_N \notin L$. Sei nun τ definiert wie folgt:

$$\tau(w) = \begin{cases} w_J & \text{, falls } w \in L' \\ w_N & \text{, falls } w \notin L' \end{cases}$$

Aufgabe 6

Wir können M' benutzen, um τ in polynomialer Zeit zu berechnen. Nach Konstruktion gilt ausserdem: $\tau(w) \in L \Leftrightarrow w \in L'$. Also gilt: $L' \propto L$.

Für $L \in \mathcal{P} \backslash \{\emptyset, \Sigma^*\}$ ist $L \in \mathcal{NP}$ und wir haben gezeigt, dass für alle Sprachen $L' \in \mathcal{NP}$ gilt: $L' \propto L$. Damit ist $L \in \mathcal{NPC}$ und es gilt $\mathcal{P} \setminus \{\emptyset, \Sigma^*\} = \mathcal{NPC}$ falls $\mathcal{P} = \mathcal{NP}$.

Aufgabe 6

Wir können M' benutzen, um τ in polynomialer Zeit zu berechnen. Nach Konstruktion gilt ausserdem: $\tau(w) \in L \Leftrightarrow w \in L'$. Also gilt: $L' \propto L$.
Für $L \in \mathcal{P} \backslash \{\emptyset, \Sigma^*\}$ ist $L \in \mathcal{NP}$ und wir haben gezeigt, dass für alle Sprachen $L' \in \mathcal{NP}$ gilt: $L' \propto L$. Damit ist $L \in \mathcal{NPC}$ und es gilt $\mathcal{P} \setminus \{\emptyset, \Sigma^*\} = \mathcal{NPC}$ falls $\mathcal{P} = \mathcal{NP}$.

b) Für o.g. Argumentation ist es essentiell, dass es Wörter $w_J, w_N \in \Sigma^*$ mit $w_J \in L$ und $w_N \notin L$ gibt. Sowohl für $\emptyset$ als auch für Σ^* ist dies nicht der Fall. Man kann auf diese Sprachen also nicht reduzieren.

Aufgabe 7

Betrachten Sie das Problem NEAR TAUT: Gegeben sei ein Boolescher Ausdruck A. Es ist zu entscheiden, ob es höchstens eine Belegung der Variablen gibt, so dass A falsch wird.

a) Formulieren Sie das komplementäre Problem co-NEAR TAUT.

b) Zeigen Sie, dass NEAR TAUT in **co-**$\mathcal{NP}$ liegt?

Aufgabe 7

a) Problem NEAR TAUT ist gegeben. Wir bilden das komplementäre Problem co-NEAR TAUT:
Gegeben ist ein Boolescher Ausdruck A.
Gibt es mindestens zwei Belegungen, für die A falsch wird?

Aufgabe 7

a) Problem NEAR TAUT ist gegeben. Wir bilden das komplementäre Problem co-NEAR TAUT:
Gegeben ist ein Boolescher Ausdruck A.
Gibt es mindestens zwei Belegungen, für die A falsch wird?

b) z.z. NEAR TAUT in **co-**$\mathcal{NP}$

Aufgabe 7

a) Problem NEAR TAUT ist gegeben. Wir bilden das komplementäre Problem co-NEAR TAUT:
Gegeben ist ein Boolescher Ausdruck A.
Gibt es mindestens zwei Belegungen, für die A falsch wird?

b) z.z. NEAR TAUT in **co-**$\mathcal{NP}$ $\Leftrightarrow$ co-NEAR TAUT in $\mathcal{NP}$
Um nun zu zeigen, dass co-NEAR TAUT in $\mathcal{NP}$ liegt, geben wir eine polynomial beschränkte NTM für co-NEAR TAUT an, die wie folgt arbeitet:

- Orakelphase: Schreibe zwei unterschiedliche Belegungen für A auf das Band.
- Verifikationsphase: Prüfe, ob beide Belegungen verschieden sind. Prüfe, ob A unter beiden Belegungen falsch ist. Falls ja, akzeptiere die Eingabe, ansonsten verwerfe diese.

Aufgabe 8

Zeigen Sie, dass das Optimierungsproblem TRAVELING SALESMAN $\mathcal{NP}$-schwer ist.
Hinweis: Die zu einem Entscheidungsproblem Π gehörende Relation R_Π sei gegeben durch

$$R_\Pi = \{(x, J) \mid x \in J_\Pi\} .$$

Aufgabe 8

Wir führen eine Turingreduktion durch:
Im Folgenden bezeichne TSP_E das Entscheidungsproblem und TSP_S das Suchproblem von TRAVELING SALESMAN. Das Entscheidungsproblem besteht darin, zu entscheiden ob es zu einem gegebenen Parameter k eine Tour der Länge $\leq k$ gibt. Die zu TSP_E bzw. TSP_S gehörenden Relationen R_E und R_S sind gegeben durch:

Aufgabe 8

Wir führen eine Turingreduktion durch:
Im Folgenden bezeichne TSP_E das Entscheidungsproblem und TSP_S das Suchproblem von TRAVELING SALESMAN. Das Entscheidungsproblem besteht darin, zu entscheiden ob es zu einem gegebenen Parameter k eine Tour der Länge $\leq k$ gibt. Die zu TSP_E bzw. TSP_S gehörenden Relationen R_E und R_S sind gegeben durch:
$R_E := \{(x, J) | x \in J_{TSP_E}\}$ und
$R_S := \{(x, y) | x \in D_{TSP_S}, y \in S_{TSP_S}(x)\}$.

Aufgabe 8

Wir führen eine Turingreduktion durch:
Im Folgenden bezeichne TSP_E das Entscheidungsproblem und TSP_S das Suchproblem von TRAVELING SALESMAN. Das Entscheidungsproblem besteht darin, zu entscheiden ob es zu einem gegebenen Parameter k eine Tour der Länge $\leq k$ gibt. Die zu TSP_E bzw. TSP_S gehörenden Relationen R_E und R_S sind gegeben durch:
$R_E := \{(x, J) | x \in J_{TSP_E}\}$ und
$R_S := \{(x, y) | x \in D_{TSP_S}, y \in S_{TSP_S}(x)\}$.
R_E ist im Prinzip nichts anderes als die Menge J_{TSP_E} geschrieben als Relation.
$(x, y) \in R_S \Leftrightarrow y$ eine optimale Tour für TSP_S.

Aufgabe 8

Wir zeigen: $R_E \propto R_S$. Dazu geben wir eine OTM mit Orakel $\Omega : \Sigma^* \rightarrow \Sigma^*$ an. Ω realisiert R_S (d.h. Ω rät zu einer Instanz von TSP eine optimale Tour).
Die OTM arbeitet wie folgt:

Aufgabe 8

Wir zeigen: $R_E \propto R_S$. Dazu geben wir eine OTM mit Orakel $\Omega : \Sigma^* \to \Sigma^*$ an. Ω realisiert R_S (d.h. Ω rät zu einer Instanz von TSP eine optimale Tour).
Die OTM arbeitet wie folgt:

1) Schreibe die Eingabe auf das Orakelband und gehe in den Zustand q_f.
2) Weise das Orakel an, in einem Schritt $\Omega(w)$ auf das Orakelband zu schreiben und anschliessend in den Zustand q_a zu wechseln.
3) Prüfe, ob $\Omega(w)$ eine Tour der Länge $\leq k$ kodiert. Falls ja, lösche das Band und schreibe J, andernfalls lösche das Band.

Aufgabe 8

Wir zeigen: $R_E \propto R_S$. Dazu geben wir eine OTM mit Orakel $\Omega : \Sigma^* \to \Sigma^*$ an. Ω realisiert R_S (d.h. Ω rät zu einer Instanz von TSP eine optimale Tour).
Die OTM arbeitet wie folgt:

1) Schreibe die Eingabe auf das Orakelband und gehe in den Zustand q_f.
2) Weise das Orakel an, in einem Schritt $\Omega(w)$ auf das Orakelband zu schreiben und anschliessend in den Zustand q_a zu wechseln.
3) Prüfe, ob $\Omega(w)$ eine Tour der Länge $\leq k$ kodiert. Falls ja, lösche das Band und schreibe J, andernfalls lösche das Band.

Die angegebene OTM realisiert R_E und hat polynomial beschränkte Laufzeit.

Aufgabe 9

Formulieren Sie das Problem EXACT COVER als ILP.

Aufgabe 9

Problem **EXACT COVER**

Gegeben: Eine endliche Menge X und eine Familie S von Teilmengen von X.

Frage: Existiert eine Menge $S' \subseteq S$, sodass jedes Element aus X in genau einer Menge aus S' liegt?

Aufgabe 9

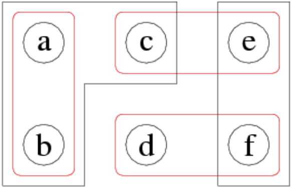
Problem **EXACT COVER**
Gegeben: Eine endliche Menge X und eine Familie S von Teilmengen von X.
Frage: Existiert eine Menge $S' \subseteq S$, sodass jedes Element aus X in genau einer Menge aus S' liegt?

EXACT COVER ist $\mathcal{NP}$-vollständig.
Beispiel:

Aufgabe 9

Problem **EXACT COVER**
Gegeben: Eine endliche Menge X und eine Familie S von Teilmengen von X.
Frage: Existiert eine Menge $S' \subseteq S$, sodass jedes Element aus X in genau einer Menge aus S' liegt?

EXACT COVER ist $\mathcal{NP}$-vollständig.
Beispiel:



Aufgabe 9

Sei $I = (X, S)$ für eine Menge $X = \{x_1, \ldots, x_n\}$ und eine Menge von Teilmengen $S = \{S_1, \ldots, S_m\}$ von X eine Instanz von EXACT COVER. Wir definieren

$$a_{ij} := \begin{cases} 1 & \text{, falls } x_i \in S_j \\ 0 & \text{, falls } x_i \notin S_j \end{cases}$$

für $1 \leq i \leq n$ und $1 \leq j \leq m$. Ausserdem seien $A = (a_{ij}) \in \{0,1\}^{n \times m}$ und $b = (1, \ldots, 1) \in \{0,1\}^n$.

Aufgabe 9

Es existiert genau dan eine Menge $S' \subseteq S$, sodass jedes Element aus X in genau einer Menge aus S' liegt, wenn es einen Vektor $x = (x_1, \ldots, x_m) \in \{0,1\}^m$ gibt, sodass

$$Ax = b$$

gilt. Der Vektor x kodiert die Mengen aus S, die in S' sind. In Matrix-Komponenten-Schreibweise ist das ILP dann gegeben als:

$$\begin{pmatrix} A \\ -A \\ E \\ -E \end{pmatrix} x \leq \begin{pmatrix} 1 \\ -1 \\ 1 \\ 0 \end{pmatrix}.$$

Formal müssen wir noch $c_i = (0, \ldots, 0) \in \mathbb{Z}^m$ spezifizieren. $\sum_{i=1}^{m} c_i x_i$ wird dann trivialerweise maximiert.

Approximationsalgorithmen mit Differenzengarantie

Vorbereitung

Definition

$\mathrm{OPT}(I)$	$:=$	Wert der/einer Optimallösung für $I \in D_\Pi$
$\mathcal{A}$	$:=$	polynomialer Approximationsalgorithmus von Π
$\mathcal{A}(I)$	$:=$	Wert der Lösung, die $\mathcal{A}$ bei Eingabe von I liefert.

Definition: Approximationsalgorithmus mit Differenzengarantie

Sei Π ein Optimierungsproblem. Ein polynomialer Algorithmus $\mathcal{A}$, der für jedes $I \in D_\Pi$ einen Wert $\mathcal{A}(I)$ liefert, mit

$$|\mathrm{OPT}(I) - \mathcal{A}(I)| \leq K$$

und $K \in \mathbb{N}_0$ konstant, heisst Approximationsalgorithmus mit Differenzengarantie oder absoluter Approximationsalgorithmus.

KNAPSACK

Gegeben: Eine Menge von “Teilen” $M = \{1, ..., n\}$,
Kosten $c_1, ..., c_n \in \mathbb{N}_0$ und Gewichte $w_1, ..., w_n \in \mathbb{N}_0$
sowie ein Gesamtgewicht $W \in \mathbb{N}_0$.

Frage: Gib $x_1, ..., x_n \in \mathbb{N}_0$ an, so dass

$$\sum_{i=0}^{n} x_i w_i \leq W \text{ und } \sum_{i=0}^{n} x_i c_i \text{ maximal ist.}$$

Satz

Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen absoluten Approximationsalgorithmus $\mathcal{A}$ für $\mathrm{KNAPSACK}$.

Satz

Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen absoluten Approximationsalgorithmus $\mathcal{A}$ für $\mathrm{KNAPSACK}$.

Beweisidee:

Man zeigt, dass ein polynomialer Approximationsalgorithmus zu einem optimalen polynomialen Algorithmus abgeleitet werden kann.

Satz

Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen absoluten Approximationsalgorithmus $\mathcal{A}$ für $\mathrm{KNAPSACK}$.

Beweisidee:
Man zeigt, dass ein polynomialer Approximationsalgorithmus zu einem optimalen polynomialen Algorithmus abgeleitet werden kann.
$\Rightarrow \mathrm{KNAPSACK} \in \mathcal{P}$

Satz

Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen absoluten Approximationsalgorithmus $\mathcal{A}$ für $\mathrm{KNAPSACK}$.

Beweisidee:

Man zeigt, dass ein polynomialer Approximationsalgorithmus zu einem optimalen polynomialen Algorithmus abgeleitet werden kann.
$\Rightarrow \mathrm{KNAPSACK} \in \mathcal{P} \Rightarrow \mathcal{P} = \mathcal{NP}$

Ausfühlicher Beweis siehe Skript 4.37.

Aufgabe 1

Zeigen Sie, dass es keinen Approximationsalgorithmus mit absoluter Gütegarantie für die Minimierungsversion des $\mathcal{NP}$-vollständigen Problems **VERTEX COVER** gibt, falls $\mathcal{P} \neq \mathcal{NP}$.

Ein Vertex-Cover eines ungerichteten Graphen $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$, so dass für jede Kante aus E mindestens einer ihrer Endknoten in V' ist.

Problem **VERTEX COVER**

Gegeben: Ein ungerichteter Graph $G = (V, E)$ und eine Konstante K.

Frage: Hat G ein Vertex-Cover der Grösse höchstens K?

Approximationsalgorithmen mit relativer Gütegarantie

Definition: Apprimationsalgorithmus mit relativer Gütegarantie

Sei Π ein Optimierungsproblem. Ein polynomialer Algorithmus $\mathcal{A}$, der für jedes $I \in D_\Pi$ einen Wert $\mathcal{A}(I)$ liefert mit $\mathcal{R}_\mathcal{A}(I) \leq K$, wobei $K \geq 1$ eine Konstante und

$$\mathcal{R}_\mathcal{A}(I) := \begin{cases} \frac{\mathcal{A}(I)}{\mathrm{OPT}(I)} & \text{falls } \Pi \text{ Minimierungsproblem} \\ \\ \frac{\mathrm{OPT}(I)}{\mathcal{A}(I)} & \text{falls } \Pi \text{ Maximierungsproblem} \end{cases}$$

heisst Approximationsalgorithmus mit relativer Gütegarantie. $\mathcal{A}$ heißt ε-approximativ, falls $\mathcal{R}_\mathcal{A}(I) \leq 1 + \varepsilon$ für alle $I \in D_\Pi$.

BIN PACKING (Optimierungsversion)

Gegeben: endliche Menge $M = \{a_1, ..., a_n\}$ mit der Gewichtsfunktion $s : M \longrightarrow (0, 1]$

Gesucht: Eine Zerlegung von M in eine minimale Anzahl von Teilmengen $B_1, ..., B_m$, so dass

$$\sum_{a_i \in B_j} s(a_i) \leq 1 \text{ für } 1 \leq j \leq m.$$

Algorithmus NEXT FIT (NF)

Eingabe : Menge M und Gewichtsfunktion s
Ausgabe : Approximationslösung für BIN PACKING

```
Füge a_1 in B_1 ein
Für a_ℓ ∈ {a_2, ... a_n}
    Sei B_j die letzte, nicht-leere Menge
    Wenn s(a_ℓ) ≤ 1 − Σ_{a_i ∈ B_j} s(a_i)
        Füge a_ℓ in B_j ein
    sonst
        Füge a_ℓ in B_{j+1} ein
```

Aufgabe 2

Zeige: **NEXT FIT** erfüllt $\mathcal{R}_{\text{NEXT FIT}} = 2$.

Algorithmus VERTEX-COVER-APPROX

```
C ← ∅
E' ← E[G]
solange E' ≠ ∅ tue
    e ← {u, v} beliebige Kante aus E'
    C ← C ∪ {u, v}
    entferne alle Kanten inzident zu u, v aus E'
return C
```

Aufgabe 3

Zeige: **VERTEX-COVER-APPROX** erfüllt $\mathcal{R}_{\mathrm{VCA}} = 2$.

Approximationsalgorithmen mit relativer Gütegarantie

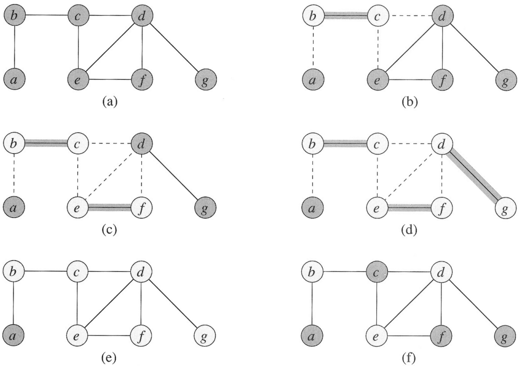


Figure 35.1 The operation of APPROX-VERTEX-COVER. **(a)** The input graph G, which has 7 vertices and 8 edges. **(b)** The edge (b, c), shown heavy, is the first edge chosen by APPROX-VERTEX-COVER. Vertices b and c, shown lightly shaded, are added to the set C containing the vertex cover being created. Edges (a, b), (c, e), and (c, d), shown dashed, are removed since they are now covered by some vertex in C. **(c)** Edge (e, f) is chosen; vertices e and f are added to C. **(d)** Edge (d, g) is chosen; vertices d and g are added to C. **(e)** The set C, which is the vertex cover produced by APPROX-VERTEX-COVER, contains the six vertices b, c, d, e, f, g. **(f)** The optimal vertex cover for this problem contains only three vertices: b, d, and e.

Algorithmus $\textsc{Vertex-Cover-Approx-2}$

$C \leftarrow \emptyset$
$E' \leftarrow E[G]$
solange $E' \neq \emptyset$ **tue**
  $v \leftarrow$ Knoten mit maximalem Grad in $G' = (V \setminus C, E')$
  $C \leftarrow C \cup \{v\}$
  entferne alle Kanten inzident zu v aus E'
return C

Aufgabe 4

Zeige: **VERTEX-COVER-APPROX-2** erfüllt $\mathcal{R}_{\text{VCA-2}} = 2$ nicht.
Hinweis: Gib ein Gegenbeispiel an!

Weitere Literatur:

Kapitel 7 in
http://i11www.ira.uka.de/teaching/WS_0506/algotech/skript/Skript.pdf

http://i11www.ira.uka.de/teaching/WS_0607/algotech/ueb/uebungsblatt06.pdf

Kapitel 35 in
INTRODUCTION TO ALGORITHMS SECOND EDITION, Cormen et al.
http://www.ubka.uni-karlsruhe.de/hylib-bin/suche.cgi?opacdb=UBKA_OPAC&nd=9184979

Reflexion

Was haben wir heute gelernt?

Reflexion

Was haben wir heute gelernt?

- Nichtexistenzbeweise von Approximationsalgorithmen mit absoluter Gütegarantie.

Reflexion

Was haben wir heute gelernt?

- Nichtexistenzbeweise von Approximationsalgorithmen mit absoluter Gütegarantie.
- Beweis der relativen Gütegarantie bei Approximationsalgorithmen mit rel. Gütegarantie.

Noch Fragen?

Vorschau

Vorschau

- Approximationsschemata

Vorschau

- Approximationsschemata
- Neues Kapitel Chomsky-0 Grammatiken

Bis zum nächsten Mal