Theoretische Informatik II

Prof. Christoph Kreitz / Nuria Brede

Universität Potsdam, Theoretische Informatik — Sommersemester 2009

Übungsblatt 9 (Version 2) — Abgabetermin: Dienstag, 30.06.09, 16:00 Uhr

---

Zur Vorbereitung ist es sinnvoll, sich mit folgenden Fragen auseinanderzusetzen:

- Wie analysiert man das Laufzeitverhalten eines Algorithmus?
- Was ist der Unterschied zwischen der Komplexität eines Problems und der Komplexität eines konkreten Lösungsverfahrens für dieses Problem?
- Was versteht man unter den Komplexitätsklassen $\mathcal{P}$ und $\mathcal{NP}$? Welche weiteren Komplexitätsklassen gibt es?
- *Praktischer Hinweis:* Im Netz findet sich eine Zusammenstellung wesentlicher Grundbegriffe der Graphentheorie. Es ist für das weitere Verständnis unerlässlich, sich mit diesen Begriffen vertraut zu machen. (▷ *http://www.cs.uni-potsdam.de/ti/lehre/downloads/TI-II/handout-graphentheorie.pdf*)

Kurzquiz 9

(1)	Ein Hamilton'scher Kreis in einem Graphen ist ein Kreis, der jede Kante des Graphen einmal enthält.	**ja**	**nein**
(2)	Das Travelling Salesman-Problem besteht darin, in einem gewichteten Graphen einen Weg zu finden, der jeden Knoten genau einmal enthält.	**ja**	**nein**
(3)	Die Zerlegung von Zahlen in ihre Primfaktoren benötigt deterministisch exponentielle Zeit.	**ja**	**nein**
(4)	Die Lösung eines Problems, das in $\mathcal{NP}$ liegt, ist in polynomieller Zeit überprüfbar.	**ja**	**nein**
(5)	Alle Probleme, die in $\mathcal{P}$ liegen, sind auch in $\mathcal{NP}$ enthalten.	**ja**	**nein**

Aufgabe 9.1 *(Formulierung von Entscheidungsproblemen)*

Betrachten Sie folgende Formulierung des $CLIQUE$-Problems:

$$CLIQUE = \{((V,E),k) \mid E \subseteq \{\{v,v'\} \mid v,v' \in V \wedge v \neq v'\} \\ \wedge\, k \in \mathbb{N} \wedge (\exists V_c \subseteq V.|V_c| \geq k \wedge (\forall v,v' \in V_c.v \neq v' \Rightarrow \{v,v'\} \in E))\}$$

Das Entscheidungsproblem, ob ein ungerichteter Graph eine Clique der Grösse k enthält, entspricht also der Frage, ob $((V,E),k)$ in dieser Menge enthalten ist. Geben Sie auch für die folgenden Probleme eine entsprechende Mengenschreibweise an.

(a) Enthält ein gerichteter Graph einen Hamilton'schen Kreis?

(b) Enthält ein ungerichteter Graph eine unabhängige Knotenmenge mit genau k Knoten?

(c) Besitzt ein zusammenhängender, gewichteter, gerichteter Graph einen aufspannenden Baum mit einem Gewicht kleiner als k?

(d) Kann den Knoten eines ungerichteten Graphen jeweils eine von k Farben so zugeteilt werden, dass allen Knotenpaaren, die durch eine Kante verbunden sind, verschiedene Farben zugeordnet sind?

Aufgabe 9.2 *(Graphentheorie)*

Eine Knotenmenge V_i in einem Graphen G heisst *Independent Set*, wenn in G keine zwei Knoten von V_i miteinander verbunden sind.
Zeigen Sie, dass V_i genau dann ein *Independent Set* in einem Graphen $G = (V, E)$ bildet, wenn ihr Komplement $V' = V - V_i$ eine Knotenüberdeckung von G bildet.

Aufgabe 9.3 *(Komplexität von Graphenalgorithmen)*

Sei $G = (V, E)$ ein ungericheter Graph.

(a) Geben Sie eine Menge K an, die das Entscheidungsproblem „Enthält G einen Kreis?" beschreibt (vergleiche Aufgabe 9.1).

(b) Geben Sie eine informale, aber präzise Beschreibung eines Entscheidungsverfahrens für K an, das eine Zeitkomplexität von $\mathcal{O}(n^k)$ für ein $k \in \mathbb{N}$ und Eingabelänge n hat.

(Haus-)Aufgabe 9.4 *(Komplexität von Sortieralgorithmen)*

Betrachten Sie folgende Pseudo-Code-Darstellung des Quicksort-Algorithmus.

```
function quicksort(L, lowerBound, upperBound) ≡
 let function partition(L, lowerBound, upperBound) ≡

   pivot:= L[(lowerBound+upperBound) div 2];
   i:= lowerBound-1;
   j:= upperBound+1;
   while true do
     do j:= j-1  while  L[j] > pivot od;
     do i:= i+1  while  L[i] < pivot od;
     if i < j then
          aux := L[i];
          L[i] := L[j];
          L[j] := aux
        else
          return j
     fi
   od;

 if (lowerBound < upperBound) then
   q:= partition(L, lowerBound, upperBound);
   quicksort(L, lowerBound, q);
   quicksort(L, q+1, upperBound);
 fi

 return L
```

(a) Analysieren Sie die worst-case Komplexität des Programms und geben Sie die Wachstumsklasse in $\mathcal{O}$-Notation an. Überlegen Sie sich dabei zunächst, wie die Wahl des Pivot-Elements den Rekursionsbaum beeinflusst.

(b) Nehmen Sie nun an, das gewählte Privot-Element teile die Liste immer im Verhältnis $\frac{1}{4}$ zu $\frac{3}{4}$. Wie wirkt sich dies auf das Laufzeitverhalten aus?

(Haus-)Aufgabe 9.5 *(Komplexität von Graphenalgorithmen)*

Definition: Unter dem Begriff *Euler-Kreis* versteht man einen Kreis in einem Graphen, der alle Kanten des Graphen genau einmal enthlt.

Sei $G = (V, E)$ ein ungericheter Graph.

(a) Geben Sie eine Menge K_e an, die das Entscheidungsproblems „Enthält G einen Euler-Kreis?" beschreibt.

(b) Geben Sie eine informale, aber präzise Beschreibung eines Entscheidungsverfahrens für K_e an, dessen Zeitkomplexität polynomiell ist.

Hinweis: Ein zusammenhängender Graph enthält genau dann einen Euler-Kreis, wenn jeder seiner Knoten einen geraden Grad hat.

(Haus-)Aufgabe 9.6 *(Komplexität von Graphenalgorithmen)*

Sei k eine beliebige, aber fest gewählte(!) natürliche Zahl mit $k > 2$.

$CLIQUE_k = \{(V, E) \mid (V, E) \text{ Graph} \wedge \exists V_c \subseteq V.|V_c| = k \wedge (\forall v, v' \in V_c.v \neq v' \Rightarrow \{v, v'\} \in E)\}$

Beschreiben Sie informell, aber präzise einen Algorithmus, der $CLIQUE_k$ in polynomieller Zeit entscheidet. Geben Sie die genaue Zeitkomplexität Ihres Verfahrens an und begründen Sie Ihre Antwort.

(Haus-)Aufgabe 9.7 *(Probleme mit beliebig großer Zeitkomplexität)*

Beweisen Sie folgende Aussage:

Sei $f : \mathbb{N} \Rightarrow \mathbb{N}$ eine total berechenbare Funktion. Dann gibt es eine entscheidbare Sprache L, die von keiner Turingmaschine M in Zeit $T_M(n) \in \mathcal{O}(f)$ entschieden werden kann.

Hinweis: Verwenden Sie Diagonalisierung als Beweismethode und benutzen Sie eine geeignete Rechenzeit in Ihrer Definition der Sprache L.

Zur Erinnerung...

In diesem Semester ist die Bearbeitung der Übungen fakultativ. Zur Selbstkontrolle ist es jedoch sinnvoll, die von Ihnen bearbeiteten Übungen korrigieren zu lassen (in diesem Fall gilt der Abgabetermin).

Sprechzeiten:

Ganz wichtig und wirklich ernst gemeint: Falls Sie Schwierigkeiten haben, sprechen Sie uns an! Auch wenn es etwas Überwindung kostet, Fragen hilft!

- Sprechen Sie in den Übungen Ihre Tutorin, bzw. Ihren Tutor an.
- Prof. Dr. Christoph Kreitz, Raum 1.18, kreitz@cs.uni-potsdam.de, Tel. (0331) 977 3060,
  Sprechstunde: mittwochs 9.30-10.30 Uhr und immer, wenn die Tür des Raumes 1.18 offen steht
- Nuria Brede, Raum 1.24, brede@cs.uni-potsdam.de, Tel. (0331) 977 3071,
  Sprechstunde: donnerstags 13-14 Uhr und nach Vereinbarung