Theoretische Informatik II

Prof. Dr. Christoph Kreitz / Holger Arnold
Universität Potsdam, Theoretische Informatik, Sommersemester 2006

Übung 13 (Version 1)

Quiz 13

Markieren Sie die folgenden Aussagen als wahr (w) oder falsch (f).

[ ] Ein Problem ist pseudopolynomiell, wenn es polynomiell in der Größe der Eingabewerte (im Gegensatz zur Länge ihrer Darstellung) ist.

[ ] Ein stark NP-vollständiges Problem kann auch pseudopolynomiell sein.

[ ] Für jedes NP-harte Optimierungsproblem lassen sich beliebig genaue Näherungslösungen effizient bestimmen.

Aufgabe 13.1

Beweisen Sie durch Diagonalisierung folgende vereinfachte Variante des Zeithierarchiesatzes: $\mathrm{TIME}(n) \neq \mathrm{TIME}(n^2)$. Konstruieren Sie dazu eine TM mit Zeitkomplexität $O(n^2)$, die sich anders verhält als jede TM mit Zeitkomplexität $O(n)$. Beweisen Sie diese Eigenschaft. Beachten Sie, dass die simulierende TM die Schrittzahl der simulierten TM überwachen muss. Bestimmen Sie auch auf die Komplexität dieser „Überwachung“.

Aufgabe 13.2

Betrachten Sie das Problem, eine minimale Knotenüberdeckung eines Graphen zu finden. Zeigen Sie, dass sich dieses Problem effizient approximieren lässt. Analysieren Sie dazu den folgenden Algorithmus.

approx-min-vc(G):

> Wiederhole, bis alle Kanten in G eine markierte Kante berühren:
>
> > Wähle eine Kante in G, die noch keine markierte Kante berührt und markiere diese Kante.
>
> Gib die Menge aller Knoten zurück, die Endpunkt einer markierten Kante sind.

Zeigen Sie, dass dieser einfache Algorithmus eine Knotenüberdeckung berechnet, die höchstens doppelt so viele Knoten wie die minimale Knotenüberdeckung enthält.

Aufgabe 13.3

Betrachten Sie das Travelling Salesman Problem für Graphen, deren Kostenfunktion die Dreiecksungleichung erfüllt. Analysieren Sie den folgenden Algorithmus.

approx-min-tsp(G, c):

> Wähle einen Knoten s aus G als Startknoten.
>
> Berechne einen minimalen spannenden Baum T für (G, c) mit s als Wurzel (c ist die Kostenfunktion).
>
> Gib die Liste der Knoten von G in der Reihenfolge einer Preorder-Traversierung von T zurück (d.h. besuche nacheinander die Wurzel eines Teilbaumes, dann den linken Teilbaum, dann den rechten Teilbaum).

Beweisen Sie folgende Aussagen über diesen Algorithmus:

1. Das Gewicht eines minimalen spannenden Baumes von G ist eine untere Schranke für die Länge der kürzesten Tour durch G.

2. Die Länge der von approx-min-tsp(G, c) berechneten Tour ist höchstens doppelt so groß wie das Gewicht eines minimalen spannenden Baumes von G.

3. Die von approx-min-tsp(G, c) berechnete Tour ist höchstens doppelt so lang wie die minimale Tour durch G.