Institut für Theoretische Informatik
Lehrstuhl Prof. Dr. D. Wagner

# Übungsblatt 7

Vorlesung Algorithmentechnik im WS 09/10

**Ausgabe** 21. Januar 2010
**Abgabe** 04. Februar 2010, 15:30 Uhr in der Übung (oder im Kasten vor Zimmer 319, Geb. 50.34)

Bitte schreiben Sie Ihren Namen und Ihre E-Mailadresse aber keine Matrikelnummer auf Ihr Übungsblatt. Die Bearbeitung in Zweiergruppen ist ausdrücklich erwünscht.

## Problem 1: Minimale Schnittbasis – Approximationsalgos relativer Gütegarantie [vgl. Kap. 7.1]

Der *Kantenraum* $\mathcal{E}$ eines ungerichteten, zusammenhängenden Graphen $G = (V, E)$ sei der Vektorraum aller Teilmengen der Kantenmenge $E$ von $G$ über dem Körper $GF(2)$ mit symmetrischer Differenz als Vektoraddition. Desweiteren sei ein Schnitt im Graphen $G$ definiert durch die Menge $D \subseteq E$ der Kanten, welche diesen Schnitt kreuzen. Die Menge $\mathcal{C}^*$ aller Schnitte von G (inklusive dem leeren Schnitt) ist ein Untervekrorraum von $\mathcal{E}$ (ohne Beweis). Die Kosten einer Basis $B = \{b_1, \ldots, b_d\}$ von $\mathcal{C}^*$ seien definiert als $c(B) := \sum_{i=1}^{d} c(b_i)$ mit $c(b_i)$ Anzahl der Kanten, die Schnitt $b_i$ kreuzen.

(a) Formulieren Sie die Partitionendarstellung des Schnitts $s_3 := s_1 \oplus s_2$ (mit $s_1, s_2 \in C^*$, $s_1 := (S, V \setminus S), s_2 := (T, V \setminus T)$) in Abhängigkeit der Schnittseiten $S$ und $T$.

(b) Zeigen Sie: Für je zwei Knoten $u, v \in V$ und jede Basis $B$ von $\mathcal{C}^*$ gilt, dass mindestens ein Schnitt aus $B$ die Konten $u$ und $v$ trennt.

(c) Zeigen Sie: Untenstehender Algorithmus ist ein polynomieller, relativer 2-Approximationsalgorithmus für das Optimierungsproblem Min-Schnitt-Basis, welches eine minimal gewichtete Basis von $\mathcal{C}^*$ sucht. (Hinweis: Setzen Sie voraus, dass die Ausgabe $B'$ tatsächlich eine Basis von $\mathcal{C}^*$ ist).

**Algorithmus 1** : Approx-Min-Schnitt-Basis

**Eingabe** : Graph $G = (V, E)$
**Ausgabe** : Basis $B'$ des Schnittraumes $\mathcal{C}^*$ von $G$

```
1 Wähle einen Knoten v ∈ V
2 B' ← ∅
3 forall v' ∈ V \ {v} do
4 |   b_{v'} ← {e ∈ E | e inzident zu v'}
5 |   B' ← B' ∪ {b_{v'}}
6 Return B'
```

## Problem 2: Approx-Subset-Sum – FPAS [vgl. Kapitel 7.2 im Skript]

Das Optimierungs-Subset-Sum-Problem sucht zu einer Menge $S := \{x_1, \ldots, x_n\} \subset \mathbb{N}$ und einem Wert $t \in \mathbb{N}$ eine Teilmenge $M \subseteq S$, so dass die Summe $z$ aller Elemente in $M$ maximal, aber nicht

größer als $t$, ist. Der folgende Algorithmus gibt abhängig von $\epsilon$ einen Wert $z' \leq t$ einer Teilmengensumme aus. Zeigen Sie: Dieser Algorithmus ist ein FPAS für den Wert $z$ einer optimalen Lösung von SUBSET-SUM (Bemerkung: Ohne Zeile 5 liefert der Algorithmus einen optimalen Wert $z$).

**Algorithmus 2** : APPROX-SUBSET-SUM$(S, t, \epsilon)$

**Ausgabe** : Teilmengensumme $z'$

1 $n \leftarrow |S|$
2 $L_0 \leftarrow \langle 0 \rangle$
3 **for** $i = 1, \ldots, n$ **do**
4 $\quad L_i \leftarrow$ MERGE-LISTS$(L_{i-1}, L_{i-1} + x_i)$ `// addiert` $x_i$ `zu jedem Element`
`// MERGE vereinigt Listen unter`
`// Erhaltung der aufsteigenden Sortierung`
5 $\quad L_i \leftarrow$ TRIM$(L_i, \epsilon/2n)$
6 $\quad$ Entferne alle Elemente größer $t$ aus $L_i$
7 Return größtes (letztes) Element $z'$ in $L_n$

**Algorithmus 3** : TRIM$(L := \langle y_1 \ldots, y_m \rangle, \delta)$

**Ausgabe** : verkürzte Liste $L'$

1 $L' \leftarrow \langle y_1 \rangle$
2 $last \leftarrow y_1$
3 **for** $i = 2, \ldots, m$ **do**
4 $\quad$ **if** $y_i > last \cdot (1 + \delta)$ **then**
`// ` $y_i \geq last$`, da` $L$ `sortiert`
5 $\quad\quad$ Füge $y_i$ an $L'$ an
6 $\quad\quad$ $last \leftarrow y_i$
7 Return $L'$

**Problem 3:** Gleichverteiltes JA/NEIN – Randomisierte Algorithmen [vgl. Kapitel 8 im Skript]

Gegeben sei ein Algorithmus BIASED-RANDOM der mit Wahrscheinlichkeit $p$ den Wert 1 ausgibt und mit $(1-p)$ den Wert 0. Geben Sie einen Algorithmus an, der BIASED-RANDOM als Methode benutzt und jeweils mit Wahrscheinlichkeit 1/2 den Wert 0 oder 1 ausgibt. Analysieren Sie außerdem die erwartete Laufzeit Ihres Algorithmus in Abhängigkeit von $p$.