Komplexitätstheorie

Sommersemester 2020

Mario Holldack, M. Sc.
Prof. Dr. Georg Schnitger
Hannes Seiwert, M. Sc.

GOETHE UNIVERSITÄT
FRANKFURT AM MAIN

Institut für Informatik
AG Theoretische Informatik

Übungsblatt 1

Ausgabe: 27.04.2020
Abgabe: 04.05.20 bis 12 Uhr

- Für jedes Übungsblatt gilt: Alle Antworten sind mathematisch fundiert zu begründen, außer der Aufgabentext erlaubt, dass eine Begründung entfallen darf.
- Durch die Übungen können Sie eine Bonifikation für die mündliche Prüfung erwerben: Bei Erreichen von 50% bzw. 70% der möglichen Übungspunkte kann die Prüfungsnote um einen bzw. zwei Notenschritte verbessert werden, falls die Prüfung bestanden ist.
- Die Abgabe des Übungsblattes erfolgt per E-Mail an holldack@em.uni-frankfurt.de. Fassen Sie Ihre Abgabe bitte in genau einer PDF-Datei zusammen und gestalten Sie den Hintergrund weiß, sodass die Abgabe gut lesbar ist. Apps wie etwa Open Note Scanner helfen Ihnen dabei. LaTeX-Abgaben sind stets willkommen.
- **Zwei aus drei.** Wählen Sie aus den Aufgaben 1.1, 1.2 und 1.3 zwei Aufgaben zur Bewertung aus. Die dritte Aufgabe fließt nicht in die Bewertung ein. Sie können bis zu 24 Punkte erreichen.

Aufgabe 1.1 *Faktorisierung* (4 + 4 + 4 = 12 Punkte)

Die Sprache FAKTORISIERUNG besteht aus allen Paaren $(N, k) \in \mathbb{N}^2$, so dass die natürliche Zahl N einen Primfaktor der Größe höchstens k besitzt. Die Zahlen N und k sind hierbei in ihrer Binärdarstellung gegeben. Zeigen Sie:

a) Wenn FAKTORISIERUNG $\in$ P, dann kann jede Zahl in polynomieller Zeit faktorisiert werden.

b) FAKTORISIERUNG $\in$ NP $\cap$ coNP.

c) Wenn FAKTORISIERUNG NP-vollständig ist, dann gilt NP = coNP.

Hinweis: Sie dürfen den AKS-Primzahltest als Blackbox benutzen.

Fazit: Man vermutet, dass NP $\neq$ coNP *gilt. Daher ist* FAKTORISIERUNG *vermutlich nicht* NP*-vollständig.*

Aufgabe 1.2 *Exaktes Independent Set* (4 + 4 + 4 = 12 Punkte)

ExactIndependentSet ist das wie folgt definierte Entscheidungsproblem: Gegeben ein Graph $G = (V, E)$ und eine natürliche Zahl $k \in \mathbb{N}_{>0}$, entscheide, ob die größte unabhängige Knotenmenge die Kardinalität genau k hat.

Zeigen Sie: ExactIndependentSet $\in \Sigma_2^p \cap \Pi_2^p$.

Bitte wenden!

Gehen Sie hierfür wie folgt vor: Um $L \in \Sigma_2^p$ bzw. $L \in \Pi_2^p$ für eine Sprache L nachzuweisen, genügt es, für jede Eingabeinstanz eine Formel von der Form $\exists x \, \forall y \, \alpha(x, y)$ bzw. $\forall x \, \exists y \, \beta(x, y)$ mit $x = (x_1, \ldots, x_k)$ und $y = (y_1, \ldots, y_\ell)$ anzugeben, wobei der „Wahrheitswert“ des Prädikats $\alpha(x, y)$ bzw. $\beta(x, y)$ deterministisch in polynomieller Zeit bestimmt werden kann.

a) Formalisieren Sie die Aussage „Der Graph G hat eine unabhängige Knotenmenge der Kardinalität genau k“ durch eine Formel φ_a.

b) Formalisieren Sie die Aussage „In G gibt es keine unabhängige Knotenmenge der Kardinalität $k + 1$“ durch eine Formel φ_b.

Begründen Sie jeweils, weshalb die in φ_a und φ_b verwendeten Prädikate effizient auswertbar sind. Folgern Sie schließlich:

c) ExactIndependentSet $\in \Sigma_2^p \cap \Pi_2^p$.

Hinweis: Die Formeln in a) und b) benötigen keine Alternationen.

Aufgabe 1.3 *Smoothed Complexity und Pseudopolynomialität* (6 + 6 = 12 Punkte)

Wir zeigen in dieser Aufgabe einen fundamentalen Zusammenhang zwischen polynomieller geglätteter Komplexität und der Existenz pseudopolynomieller Algorithmen für binäre Optimierungsprobleme von der Form

$$\max \left\{ \sum_{i=1}^{n} c_i x_i \;:\; x \in L \subseteq \{0,1\}^n \right\}, \qquad (*)$$

wobei $c = (c_1, \ldots, c_n) \in \mathbb{N}^n$ der Kostenvektor und L die Menge aller zulässigen Lösungen ist.

In der Vorlesung haben wir bereits geglättete Komplexität für gaußsches Rauschen kennengelernt. Wir betrachten in dieser Aufgabe uniform verteiltes Rauschen.

Sei A ein Algorithmus für $(*)$ mit polynomieller geglätteter Komplexität und sei A' der wie folgt beschriebene randomisierte Algorithmus mit Eingabe c:

1. Für $c_{\max} := \max\{c_1, \ldots, c_n\}$ erhalte den normierten Kostenvektor $c' := \frac{1}{c_{\max}} \cdot (c_1, \ldots, c_n)$.
2. Verrausche jede Komponente c'_i von c' durch einen uniform auf $[0, \frac{1}{n \cdot c_{\max}}]$ verteilten Fehlerterm δ_i, d. h. der verrauschte Kostenvektor ist $c'' := (c'_1 + \delta_1, \ldots, c'_n + \delta_n)$.
3. Führe A mit Eingabe c'' aus.
4. Gib die von A ermittelte Lösung $x^{(A)} := \arg\max \left\{ \sum_{i=1}^{n} c''_i x_i \;:\; x \in L \subseteq \{0,1\}^n \right\}$ aus.

Zeigen Sie:

a) Die erwartete Laufzeit von A' ist pseudopolynomiell, d. h. polynomiell in n und $c_{\max}$.

Hinweis: Sie dürfen annehmen, dass A' über einen Zufallszahlengenerator verfügt, der die uniform verteilten Fehlerterme $\delta_1, \ldots, \delta_n$ erzeugt.

b) Der Algorithmus A' ist korrekt, d. h. für jede Eingabe c liefert A' die gleiche Ausgabe wie A.

Hinweis: Sie dürfen annehmen, dass $(*)$ genau eine optimale Lösung besitzt.

Fazit: Wenn es einen Algorithmus A mit polynomieller geglätteter Komplexität für $(*)$ gibt, dann gibt es einen Algorithmus A' für $(*)$ mit pseudopolynomieller erwarteter Laufzeit. Man kann sogar zeigen, dass auch die umgekehrte Richtung gilt:

Satz. *Das Maximierungsproblem $(*)$ kann genau dann mit polynomieller geglätteter Laufzeit gelöst werden, wenn es durch einen randomisierten Algorithmus in pseudopolynomieller erwarteter Laufzeit gelöst werden kann.*

Korollar. *Das* `NP`*-vollständige Rucksackproblem hat polynomielle geglättete Komplexität.*