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

Übungsblatt 4

Vorlesung Theoretische Grundlagen der Informatik im WS 20/21

Ausgabe: 23. Dezember 2020
Abgabe: 12. Januar 2021, 11:30 Uhr (digital im ILIAS)

Aufgabe 1 (3 Punkte)

In der Vorlesung wurden zwei nichtdeterministische Varianten der Turing-Maschine vorgestellt: Die klassische nichtdeterministische Turing-Maschine (NTM) hat Wahlmöglichkeiten in der Übergangsfunktion, analog zum nichtdeterministischen endlichen Automaten. Die Orakel-Turing-Maschine (OTM) lagert den Nichtdeterminismus in ein Orakelmodul aus, das zu Beginn der Berechnung ein Orakelwort vor die Eingabe schreibt. Danach wird deterministisch weitergearbeitet.

In Übung 5 haben wir die Klasse ONP definiert, die aus allen Problemen besteht, die von einer OTM in polynomialer Zeit entschieden werden können. Wir haben gezeigt, dass $\mathsf{NP} \subseteq \mathsf{ONP}$ gilt. Zeigen Sie nun: $\mathsf{ONP} \subseteq \mathsf{NP}$.

Aufgabe 2 (5 Punkte)

In der Vorlesung wurde (ohne Beweis) behauptet, dass eine universelle Turing-Maschine existiert, die als Eingabe eine Gödelnummer w und ein Wort v bekommt und dann die Ausführung der Turing-Maschine T_w mit der Eingabe v simuliert. In dieser Aufgabe sollen Sie konkret die Arbeitsweise einer solchen universellen Turing-Maschine T_U beschreiben. Ihre universelle Turing-Maschine darf mehrere Bänder verwenden. Der Einfachheit halber können Sie davon ausgehen, dass ein festes Bandalphabet Γ vorgegeben ist, das sowohl T_U als auch alle von T_U simulierten Turing-Maschinen verwenden. Die Laufzeit Ihrer Simulation ist unerheblich.

Ihre Beschreibung von T_U sollte mindestens folgende Fragen beantworten:

- Wieviele Bänder hat T_U und wofür werden sie benutzt?
- Wie repräsentiert T_U die aktuelle Bandbeschriftung von T_w?
- Wie repräsentiert T_U den aktuellen Zustand der endlichen Kontrolle von T_w?
- Wie repräsentiert T_U die aktuelle Kopfposition von T_w?
- Wie identifiziert T_U den passenden Übergang von T_w für den aktuellen Zustand und die aktuelle Bandbeschriftung?
- Wie führt T_U diesen Übergang aus, d.h. wie aktualisiert sie Bandbeschriftung, Kopfposition und Zustand?

Aufgabe 3 (4 Punkte)

Die Komplexitätsklasse EXP ist definiert als die Menge aller Entscheidungsprobleme, die in deterministisch exponentieller Zeit gelöst werden können, d.h. in Zeit $\mathcal{O}(2^{n^c})$ für eine Konstante c. Analog dazu enthält die Klasse NEXP genau die Probleme, die sich in nichtdeterministisch exponentieller Zeit lösen lassen.

Wir wollen nun analog zur NP-Schwere den Begriff der NEXP-Schwere einführen. Dazu benötigen wir den Begriff der NEXP-Transformation. Eine NEXP-Transformation von einem Problem Π_1 in ein Problem Π_2 ist eine Funktion $f_{\mathsf{NEXP}} \colon D_{\Pi_1} \to D_{\Pi_2}$. Wie bei der polynomiellen Transformation fordern wir, dass eine Instanz $I \in D_{\Pi_1}$ genau dann eine Ja-Instanz von Π_1 ist, wenn $f_{\mathsf{NEXP}}(I)$ eine Ja-Instanz von Π_2 ist. Bei der polynomiellen Transformation wurde zusätzlich noch gefordert, dass f_{NEXP} in deterministisch polynomieller Zeit berechnet werden kann. In dieser Aufgabe sollen Sie untersuchen, ob und wie wir diese Forderung ändern müssen.

Falls eine NEXP-Transformation von Π_1 in Π_2 existiert, schreiben wir $\Pi_1 \propto_{\mathsf{NEXP}} \Pi_2$. Wir nennen ein Problem Π NEXP-schwer, wenn $\Pi' \propto_{\mathsf{NEXP}} \Pi$ für jedes Problem $\Pi' \in \mathsf{NEXP}$ gilt. In der Vorlesung wurde gezeigt: Wenn es ein NP-schweres Problem gibt, das in P liegt, dann gilt $\mathsf{P} = \mathsf{NP}$. Eine analoge Eigenschaft wollen wir auch für NEXP-Schwere haben: Wenn es ein NEXP-schweres Problem gibt, das in EXP liegt, dann gilt $\mathsf{EXP} = \mathsf{NEXP}$.

Betrachten Sie folgende möglichen Forderungen an f_{NEXP}:

(a) f_{NEXP} kann in deterministisch polynomieller Zeit berechnet werden.

(b) f_{NEXP} kann in deterministisch polynomiellem Platz berechnet werden.

(c) f_{NEXP} kann in deterministisch exponentieller Zeit berechnet werden können.

Für welche dieser Forderungen hat NEXP-Schwere die gewünschte Eigenschaft? Begründen Sie!

Aufgabe 4 (2 + 1 + 2 = 5 Punkte)

Betrachten Sie folgende Funktion:

$$T_{\max}(n) = \max\left(\left\{ m \in \mathbb{N} \;\middle|\; \begin{array}{l}\text{Es gibt eine TM } \mathcal{M} \text{ mit } n \text{ Zuständen und ein } x \in \Sigma^*, \\ \text{sodass } \mathcal{M} \text{ bei Eingabe } x \text{ in } m \text{ Schritten hält.}\end{array} \right\}\right)$$

Beachten Sie, dass nicht haltende Berechnungen nicht in $T_{\max}$ einfließen. Also ist $T_{\max}$ die Länge der längsten *haltenden* Berechnung, die mit n Zuständen möglich ist.

Die (starke) Goldbach-Vermutung besagt, dass jede gerade Zahl größer als 2 die Summe zweier Primzahlen ist. Bisher konnte dies nicht bewiesen werden.

(a) Zeigen Sie, dass eine TM existiert, die genau dann hält, wenn die Goldbach-Vermutung falsch ist.

(b) Tatsächlich existiert eine solche TM $\mathcal{M}_G$, die 27 Zustände benötigt. Angenommen, $T_{\max}(27)$ wäre bekannt. Geben Sie ein Verfahren an, dass dann die Goldbach-Vermutung in endlicher Zeit beweist oder widerlegt.

(c) Zeigen Sie, dass $T_{\max}$ nicht berechenbar ist.

Aufgabe 5 (4 Punkte)

Für $k \in \mathbb{N}$ ist ein Graph $G = (V, E)$ k-färbbar, wenn eine Funktion $\varphi : V \to \{1, 2, \ldots, k\}$ existiert, so dass für alle $(u, v) \in E$ gilt, dass $\varphi(u) \neq \varphi(v)$. Beim Entscheidungsproblem k-Color ist gefragt, ob ein gegebener Graph G k-färbbar ist. Zeigen Sie, dass 2020-Color NP-vollständig ist. Sie dürfen benutzen, dass 3-Color NP-vollständig ist.

Aufgabe 6 (2 + 2 = 4 Punkte)

Gegeben sei ein Entscheidungsproblem Π, das mindestens eine Ja-Instanz und eine Nein-Instanz hat. Wir definieren das Entscheidungsproblem Π^* wie folgt: Die Instanzen von Π^* sind Paare (I_1, I_2) von Π-Instanzen mit der Eigenschaft, dass genau eine davon eine Ja-Instanz von Π ist. Es gilt zu entscheiden, ob I_1 die Ja-Instanz ist.

Wir definieren außerdem das Entscheidungsproblem $\overline{\Pi}$, dessen Instanzen beliebige Paare (I_1, I_2) von Π-Instanzen sind. Es soll entschieden werden, ob (I_1, I_2) eine gültige Instanz von Π^* ist, also ob genau eine der beiden Instanzen eine Ja-Instanz von Π ist.

Zeigen Sie:

(a) Wenn Π in NP liegt, dann liegt Π^* in $\mathsf{NP} \cap \text{co-}\mathsf{NP}$.

(b) Wenn Π in NPC liegt, dann ist $\overline{\Pi}$ NP-schwer.

Aufgabe 7 (5 Punkte)

Das Set-Splitting-Problem ist wie folgt definiert: Gegeben sind eine Grundmenge S und eine Menge $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ von Teilmengen $A_i \subseteq S$. Gesucht ist eine Partition $S_0 \dot{\cup} S_1 = S$, sodass kein A_i vollständig in S_0 oder S_1 enthalten ist.

Zeigen Sie, dass Set-Splitting NP-vollständig ist. Nutzen Sie 3-SAT zur Reduktion.