Hausaufgabenblatt

Hinweise:

- Die Hausaufgabe kann bis zum **14.01.2022, 10 Uhr** auf ISIS hochgeladen werden.
- Die Hausaufgabe muss in Dreiergruppen bearbeitet werden.
- Bitte verwenden Sie die LaTeX-Vorlage auf ISIS für Ihre Abgabe.
- Plagiate werden nicht toleriert und werden scharf geahndet.
- Es können bis zu **25 Portfoliopunkte** erreicht werden.
- Alle Antworten sind zu begründen. Antworten ohne Begründung erhalten **0 Punkte**. Achten Sie insbesondere darauf, die Korrektheit Ihrer angegebenen Lösungen (Programme, Reduktionen etc.) zu begründen.
- Bitte versuchen Sie, insgesamt nicht mehr als fünf Seiten (im vorgegebenen Stil) zu schreiben.

Aufgabe 1. *Eigenschaften berechenbarer Funktionen* 8 P.

Für zwei totale Funktionen $f, g\colon \mathbb{N} \to \mathbb{N}$ sagen wir, dass

- f *größer* als g ist, falls $f(n) > g(n)$ für alle $n \in \mathbb{N}$,
- f *fast überall größer* als g ist, falls ein $n_0 \in \mathbb{N}$ existiert, sodass $f(n) > g(n)$ für alle $n \geq n_0$,
- f *fast gleich* g ist, falls ein $n_0 \in \mathbb{N}$ existiert, sodass $f(n) = g(n)$ für alle $n \geq n_0$ und
- f *oft gleich* g ist, falls unendlich viele $n \in \mathbb{N}$ existieren, sodass $f(n) = g(n)$.

Zeigen oder widerlegen Sie die folgenden Aussagen:

(a) Es gibt eine (nicht notwendigerweise berechenbare) totale Funktion, die größer als alle berechenbaren totalen Funktionen ist.

(b) Es gibt eine (nicht notwendigerweise berechenbare) totale Funktion, die fast überall größer als alle berechenbaren totalen Funktionen ist.

(c) Es gibt eine berechenbare totale Funktion $f\colon \mathbb{N} \to \mathbb{N}$ und eine unberechenbare totale Funktion $g\colon \mathbb{N} \to \mathbb{N}$, sodass f und g fast gleich sind.

(d) Es gibt eine berechenbare totale Funktion $f\colon \mathbb{N} \to \mathbb{N}$ und eine unberechenbare totale Funktion $g\colon \mathbb{N} \to \mathbb{N}$, sodass f und g oft gleich sind.

Aufgabe 2. *Starke Turing-Maschinen* 9 P.

Eine *starke Turing-Maschine* ist wie eine normale deterministische Turing-Maschine ein Siebentupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$. Wie bei einer normalen Turing-Maschine ist Z eine endliche Zustandsmenge, Σ das endliche Eingabealphabet, $\delta\colon (Z \setminus E) \times \Gamma \to_p Z \times \Gamma \times \{L, R, N\}$ eine partielle Überführungsfunktion, $z_0 \in Z$ ein Startzustand, $\square \in \Gamma \setminus \Sigma$ das Blanksymbol und $E \subseteq Z$ die Menge der Endzustände. Anders als bei einer normalen Turing-Maschine ist Γ kein endliches Bandalphabet, sondern $\Gamma = \mathbb{N} \cup \Sigma \cup \{\square\}$. Dabei gehen wir davon aus, dass $\Sigma \cap \mathbb{N} = \emptyset$. Die Begriffe Konfiguration, Konfigurationsfolge, Berechenbarkeit einer Funktion und Entscheidbarkeit einer Sprache sind analog zu gewöhnlichen Turing-Maschinen definiert.

Beweisen Sie folgende Aussage: Für jede Sprache $L \subseteq \Sigma^*$ existiert eine starke Turing-Maschine M_L, sodass M_L auf jeder Eingabe hält und $T(M_L) = L$ gilt. Sie dürfen dabei annehmen, dass $\Sigma = \{a, b\}$.

Aufgabe 3. *Eingeschränktes WHILE* 8 P.

Wir definieren eine neue Programmiersprache MinWHILE zur Berechnung von Funktionen $\mathbb{N}^k \to \mathbb{N}$.

Die Syntax von MinWHILE ist identisch zu der von WHILE mit folgenden zwei Einschränkungen:

1. Es dürfen nur Zuweisungen der Form $x_i \coloneqq x_i \pm c$ und keine der Form $x_i \coloneqq x_j \pm c$ mit $i \neq j$ verwendet werden.
2. Es darf keine LOOP-Schleife verwendet werden.

Die Semantik von MinWHILE ist identisch zu der von WHILE.

Zeigen Sie, dass MinWHILE Turing-mächtig ist, indem Sie beweisen, dass jedes WHILE-Programm durch ein MinWHILE-Programm simuliert werden kann.

Notation. Für ein WHILE- beziehungsweise ein MinWHILE-Programm Q sei N_Q die größte Zahl, sodass x_{N_Q} im Programm Q benutzt wird. Das bedeutet, dass Q nur die Variablen $x_0, x_1, \ldots, x_{N_Q}$ benutzt.