Prof. T. Nipkow
K. Kappelmann, J. Rädle, L. Stevens

KEINE ABGABE

Technische Universität München
Lehrstuhl für Logik und Verifikation

Einführung in die Theoretische Informatik

Sommersemester 2020 – Übungsblatt 7

AUFGABE 7.1. (*Wichtige Begriffe*) Stufe A

Überprüfen Sie, dass Sie die folgenden Begriffe korrekt definieren können.

- nicht-deterministische/deterministische Turing-Maschine
- Konfiguration einer Turing-Maschine
- akzeptierte Sprache einer Turing-Maschine
- Turing-berechenbar

Turing Maschinen interaktiv

Es gibt verschiedene Websiten auf denen Turing Maschinen interaktiv konstruiert und simuliert werden können, z.B. https://wimmers.github.io/turing-machine-viz/. Beachten Sie dass diese Seite Endzustände nicht visualisiert.

AUFGABE 7.2. (*TM für Sprache*) Stufe C

Geben Sie für die angegebene Sprache eine passende TM M an.

(a) $\mathsf{L}_F(\mathsf{M}) = \{\mathsf{a}^n\mathsf{b}^n\mathsf{c}^n \mid n \in \mathbb{N}\}$

AUFGABE 7.3. (*TM Berechnung*) Stufe B

Konstruieren Sie eine Turingmaschine $T = (Q, \Sigma, \Gamma, q_0, \square, \delta, F)$, mit $\Sigma = \{|\}$, die eine eingegebene Strichzahl verdoppelt.

Definition (Alternative Akzeptanzbedingungen für Turing-Maschinen)

In der Vorlesung wurde die Annahme gemacht, dass die Übergangsfunktion δ einer Turingmaschine folgende Eigenschaft erfüllt:

$$\delta(q, \mathsf{a}) \textit{ ist nicht definiert für alle } q \in \mathsf{F},\ \mathsf{a} \in \Gamma.$$

Sei $\mathcal{M}_A$ die Menge der Turingmaschinen, die diese Annahme erfüllen, und sei $\mathcal{M}$ die Menge aller Turingmaschinen. Es gilt somit $\mathcal{M}_A \subsetneq \mathcal{M}$.
Für $\mathsf{M} \in \mathcal{M}$ mit $\mathsf{M} = (\mathsf{Q}, \Sigma, \Gamma, \delta, \mathsf{q}_0, \square, \mathsf{F})$ definiere:

- $\mathsf{L}_F(\mathsf{M}) = \{w \in \Sigma^* \mid \exists \alpha, \beta \in \Gamma^*, f \in \mathsf{F}.\ (\varepsilon, \mathsf{q}_0, w) \to^*_{\mathsf{M}} (\alpha, f, \beta)\}$.
  (Menge der Wörter, für die die Maschine einen Endzustand irgendwann besucht.)
- $\mathsf{L}_H(\mathsf{M}) = \{w \in \Sigma^* \mid \exists \alpha, \beta \in \Gamma^*, q \in \mathsf{Q}.\ (\varepsilon, \mathsf{q}_0, w) \to^*_M (\alpha, q, \beta)$ und $\delta(q, \mathit{first}(\beta))$ ist nicht definiert$\}$.
  (Menge der Wörter, für die die Maschine hält.)
- $\mathsf{L}_{HF}(\mathsf{M}) = \{w \in \Sigma^* \mid \exists \alpha, \beta \in \Gamma^*, f \in \mathsf{F}.\ (\varepsilon, \mathsf{q}_0, w) \to^*_M (\alpha, f, \beta)$ und $\delta(f, \mathit{first}(\beta))$ ist nicht definiert$\}$.
  (Menge der Wörter, für die die Maschine in einem Endzustand hält.)

AUFGABE 7.4. (*TM Akzeptanzbedingungen*) Stufe B

Begründen Sie folgende Aussagen, indem Sie eine passende Konstruktion angeben.

(a) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}_A$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}$ mit $\mathsf{L}_{\mathsf{F}}(\mathsf{M}) = \mathsf{L}_H(\mathsf{M}')$.
(b) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}_A$ mit $\mathsf{L}_H(\mathsf{M}) = \mathsf{L}_F(\mathsf{M}')$.
(c) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}_A$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}$ mit $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_F(\mathsf{M}')$.
(d) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}_A$ mit $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_F(\mathsf{M}')$.
(e) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}$ mit $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_H(\mathsf{M}')$.

AUFGABE 7.5. (*2-PDA und TMs*) Stufe D

Ein 2-PDA ist ein PDA, der 2 Stacks zur Verfügung hat. In jedem Schritt kann der PDA in Abhängigkeit vom aktuellen Zustand, dem gelesenen Eingabezeichen und den Symbolen, die oben auf den beiden Stacks liegen, in einen neuen Zustand wechseln und jeden der Stacks, wie im Fall eines gewöhnlichen, PDAs modifizieren.

(a) Geben Sie eine formale Definition für 2-PDAs an.
(b) Geben Sie eine Sprache an, die von einem 2-PDA, aber von keinem 1-PDA akzeptiert wird. Begründen Sie (informell), wie Sie einen 2-PDA für diese Sprache bauen könnten.

(c) Beschreiben Sie informell, wie Sie einen PDA P in eine Turingmaschine M übersetzen können, sodass $\mathsf{L}_\varepsilon(P) = \mathsf{L}_F(\mathsf{M})$ gilt.
Tipp: Verwenden Sie eine TM mit zwei Bändern.

(d) Überlegen Sie wie, Sie das Verfahren aus (c) erweitern können, sodass ein 2-PDA A in eine Turingmaschine M übersetzt werden kann.

(e) Geben Sie eine formale Übersetzung von einer Turingmaschine M in einen 2-PDA A an, sodass $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_F(A)$. Beschreiben Sie dabei ihre Idee auch informell.

(f) Zeigen Sie unter Verwendung der vorherigen Ergebnisse, dass jeder k-PDA A ($k \geq 3$) von einem 2-PDA A' simuliert werden kann, d.h. $\mathsf{L}_\varepsilon(A) = \mathsf{L}_\varepsilon(A')$. Somit können beliebig viele Stacks immer durch genau zwei Stacks simuliert werden.