Prof. Dr. Heiko Röglin
Anna Großwendt, David Kübel
Institut für Informatik

Übungsblatt 2

Aufgabe 2.1: Das Alphabet ausgeben

Für $n \geq 1$ sei $\Sigma = \{a_1, a_2, \dots a_n\}$ ein beliebiges Alphabet. Konstruieren Sie eine 1-Band Turingmaschine M mit maximal 5 Zuständen, die mit leerem Band gestartet das Alphabet auf dem Band ausgibt. D.h. nach endlich vielen Schritten hält M, der Kopf zeigt auf das erste Zeichen und der Bandinhalt ist $a_1\#a_2\#\dots\#a_n$. Beschreiben Sie Idee und Zustände von M.

Aufgabe 2.2: Rastlose TMs

Eine deterministische 1-Band Turingmaschine M kann nach Definition der Vorlesung drei Arten von Kopfbewegungen ausführen: nach links (L), nach rechts (R) oder "keine Bewegung" (N). Betrachten Sie nun eine alternative Definition einer 1-Band Turingmaschine, die sich nur darin unterscheidet, dass der Kopf immer nach links oder rechts bewegt werden muss. Eine solche Turingmaschine M' führt also stets eine Kopfbewegung L bzw. R aus.

Beweisen Sie, dass jede Turingmaschine M (deren Kopf stehen bleiben darf) durch eine rastlose Turingmaschine M' (deren Kopf sich stets bewegt) simuliert werden kann. Wie groß ist die Rechenzeit von M' im Vergleich zu M höchstens?

Aufgabe 2.3: TM analysieren

Wir betrachten die Turingmaschine $M = (Q, \{0,1\}, \Gamma, \square, q_0, \bar{q}, \delta)$ mit $\Gamma = \{0, 1, \square\}$, $Q = \{q_0, q_1, q_2, q_3, q_4, \bar{q}\}$ und der Zustandsüberführungsfunktion δ, gegeben durch folgende Tabelle:

	0	1	$\square$
q_0	$(q_0, 0, R)$	$(q_0, 1, R)$	$(q_1, \square, L)$
q_1	$(q_2, \square, R)$	$(q_3, \square, R)$	$(\bar{q}, \square, R)$
q_2	$(q_4, 0, L)$	$(q_4, 0, L)$	$(q_4, 0, L)$
q_3	$(q_4, 1, L)$	$(q_4, 1, L)$	$(q_4, 1, L)$
q_4	$(q_4, 1, L)$	$(q_4, 0, L)$	$(q_1, \square, L)$

Beschreiben Sie das Verhalten von M auf einer beliebigen Eingabe $w \in \{0,1\}^*$. Erläutern Sie kurz die Bedeutung der einzelnen Zustände. Gibt es Einträge in der Tabelle, die nur der Vollständigkeit halber existieren und nie benötigt werden?

Aufgabe 2.4: LOOP-Programme

Betrachten Sie die (stark eingeschränkte) Programmiersprache LOOP. Diese erlaubt lediglich die Verwendung von Additionen, Wertzuweisungen, Zählschleifen und Subroutinen. Ein LOOP-Programm besteht aus Variablen x_1, x_2, ..., Konstanten $c \in \mathbb{N}_0$, Trennsymbolen ; und :=, Operatoren $+, -$ und den Schlüsselwörtern LOOP, DO, END.

Die Syntax eines LOOP-Programms ist definiert durch

1. **Wertzuweisungen** unter Verwendung von Variablen x_i, x_j, Konstanten $c \in \mathbb{N}_0$ und Operatoren. Das bedeutet, dass
   - $x_i := c$
   - $x_i := x_j$
   - $x_i := x_j + c$
   - $x_i := x_j - c$

   gültige LOOP-Programme sind.
2. **Reihung**, d.h. sind P_1 und P_2 LOOP-Programme, so auch $P_1; P_2$.
3. **(Endliche) Wiederholung**, d.h. falls P ein LOOP-Programm ist, so auch LOOP x_i DO P END.

Beachten Sie: Bei der Ausführung von Wertzuweisungen werden negative Werte implizit durch Nullen ersetzt. Zu Beginn einer (endlichen) Wiederholung wird die Anzahl an Durchläufen durch x_i festgelegt und ändert sich nicht, falls x_i innerhalb der Schleife verändert wird. Die Eingaben eines Programms stehen zu Beginn in den ersten Variablen, die Ausgabe(n) am Ende ebenfalls.

Eine Funktion heißt **LOOP-berechenbar**, wenn sie sich als LOOP-Programm formulieren lässt. Zeigen Sie, dass folgende Funktionen LOOP-berechenbar sind:

$$x + y, \quad x \cdot y, \quad x^y, \quad \operatorname{sg}(x) := \begin{cases} 1, & \text{falls } x > 0 \\ 0, & \text{falls } x = 0. \end{cases}$$

Aufgabe 2.5: Turingmaschinensimulator (Zusatzaufgabe)

Besuchen Sie die Homepage `https://turingmachinesimulator.com/` und wählen Sie das Beispiel "Binary palindrome" aus. Berechnen Sie mit dem Turingmaschinensimulator die Ausgabe der Turingmaschine auf der Eingabe 110111 Die Turingmaschine in diesem Beispiel entscheidet die Sprache der Palindrome über dem Alphabet $\{0, 1\}$. Das heißt, die Turingmaschine terminiert im Simulator bei Eingabe von w mit Ausgabe 1, falls das Wort w ein Palindrom ist. Andernfalls terminiert sie mit Ausgabe 0.

Entwerfen Sie ein Turingmaschinen-Programm für den Simulator, welches die Sprache $L = \{0^n 1^{2n}\}$ entscheidet.