Universität Siegen
Lehrstuhl Theoretische Informatik
Markus Lohrey

Übungsblatt 1

Aufgabe 1. Wahr oder falsch?

(a) Es gibt überabzählbar unendlich viele Wörter über einem endlichen Alphabet.

(b) Es gibt abzählbar unendlich viele berechenbare Funktionen $f\colon \mathbb{N}^k \to \mathbb{N}$.

(c) Es gibt überabzählbar unendlich viele Funktionen $f\colon \mathbb{N}^k \to \mathbb{N}$.

Lösung.

(a) **Falsch.** Über jedem endlichen Alphabet gibt es gibt nur abzählbar viele Wörter. Eine Abzählung erhält man etwa aus der längen-lexikographischen Ordnung, bei der zunächst nach der Wortlänge und bei gleicher Länge alphabetisch sortiert wird.

In den Lösungen zu den folgeden beiden Aufgabenteilen gehen wir davon aus, dass der Parameter k fest und in Aufgabenteil (c) zudem postiv ist.

(b) **Wahr.** Nach Aufgabenteil (a) gibt es abzählbar viele Maschinen bzw. Programme, die eine Funktion der Form $f\colon \mathbb{N}^k \to \mathbb{N}$ berechnen können. Die Menge solcher Funktionen ist somit selbst abzählbar. Dass diese Menge unendlich ist, sieht man etwa daran, dass jede konstante Funktion berechenbar ist.

(c) **Wahr.** Schon die Menge aller Funktionen $f\colon \mathbb{N} \to \mathbb{N}$ ist überabzählbar; siehe Folie 8 der Vorlesung. Der Beweis aus der Vorlesung lässt sich leicht auf Funktionen der Form $f\colon \mathbb{N}^k \to \mathbb{N}$ mit $k \geq 1$ verallgemeinern. Alternativ kann man feststellen, dass sich jeder Funktion $g\colon \mathbb{N} \to \mathbb{N}$ eine Funktion $f_g\colon \mathbb{N}^k \to \mathbb{N}$ mit

$$f_g(x_1, \ldots, x_k) = g(x_1)$$

zuordnen lässt, sodass $f_{g_1} = f_{g_2}$ genau dann gilt, wenn $g_1 = g_2$ gilt. Es gibt also mindestens so viele Funktionen $f\colon \mathbb{N}^k \to \mathbb{N}$, wie es Funktionen $g\colon \mathbb{N} \to \mathbb{N}$ gibt.

Aufgabe 2. Sei $M = (Q, \Sigma, \Sigma \cup \{\Box\}, \delta, z_0, \Box, \{z_2\})$ die Turingmaschine über $\Sigma = \{0, 1\}$ mit Zustandsmenge $Q = \{z_0, z_1, z_2\}$ und den folgenden Transitionen:

$$\begin{aligned} \delta(z_0, 0) &= (z_0, 0, R) & \qquad \delta(z_1, 0) &= (z_1, 0, L) \\ \delta(z_0, 1) &= (z_0, 1, R) & \qquad \delta(z_1, 1) &= (z_1, 1, L) \\ \delta(z_0, \Box) &= (z_1, 0, L) & \qquad \delta(z_1, \Box) &= (z_2, \Box, R). \end{aligned}$$

(a) Untersuchen Sie, wie sich die Turingmaschine auf den Eingaben 10, 11 und 110 verhält. Wie verhält sie sich allgemein bei Eingaben $w \in 1\{0, 1\}^* \cup \{0\}$?

(b) Wie verändert sich das Verhalten, wenn man die Transition $\delta(z_0, \Box) = (z_1, 0, L)$ durch die Transition $\delta(z_0, \Box) = (z_1, 1, L)$ ersetzt?

(c) Ändern Sie M so ab, dass sie die Funktion $f(n) = 4n + 1$ berechnet.

Lösung.

(a) Bei beliebiger Eingabe $w \in \Sigma^*$ wird zunächst der Lesekopf der Turingmaschine bis zum Ende der Eingabe bewegt, dann das darauf folgende Blanksymbol $\Box$ durch eine 0 ersetzt und schließlich der Lesekopf wieder an den Anfang der Eingabe bewegt.

Aus der Eingabe 10 wird so 100, aus 11 wird 110 und aus 110 wird 1100.

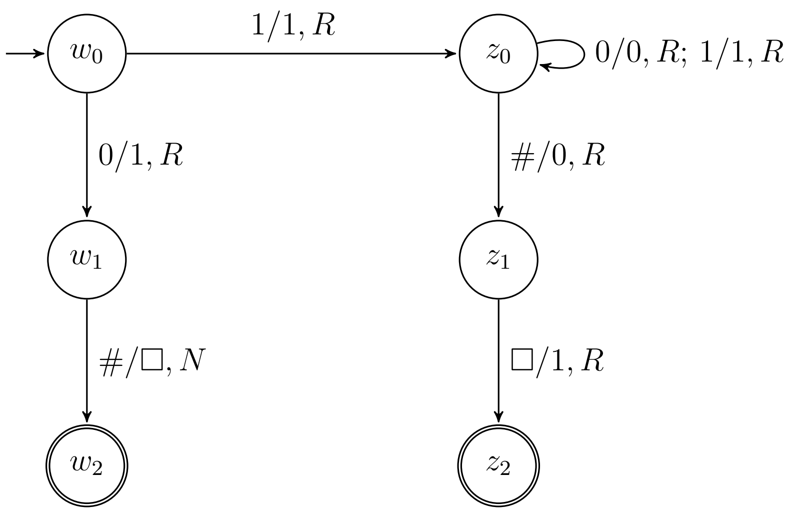
Eingaben der Form $w \in 1\{0,1\}^* \cup \{0\}$ können als binär kodierte Zahlen verstanden werden. Kodiert die Eingabe eine positive Zahl n, so steht am Ende der Berechnung die Binärkodierung von $2n$ auf dem Band, da das Anfügen einer 0 an eine Zahl im Binärsystem der Multiplikation mit 2 entspricht. Kodierung die Eingabe jedoch die Zahl $n = 0$, so erhalten wir als Ausgabe das Wort 00, was wegen der führenden 0 keine gültige Binärkodierung ist.

(b) Mit der Änderung wird am Bandende eine 1 statt einer 0 angefügt.

Interpretieren wir die Eingabe wieder als eine binär kodierte Zahl n, so steht nach der Berechnung für $n > 0$ die Binärkodierung von $2n + 1$ auf dem Band.

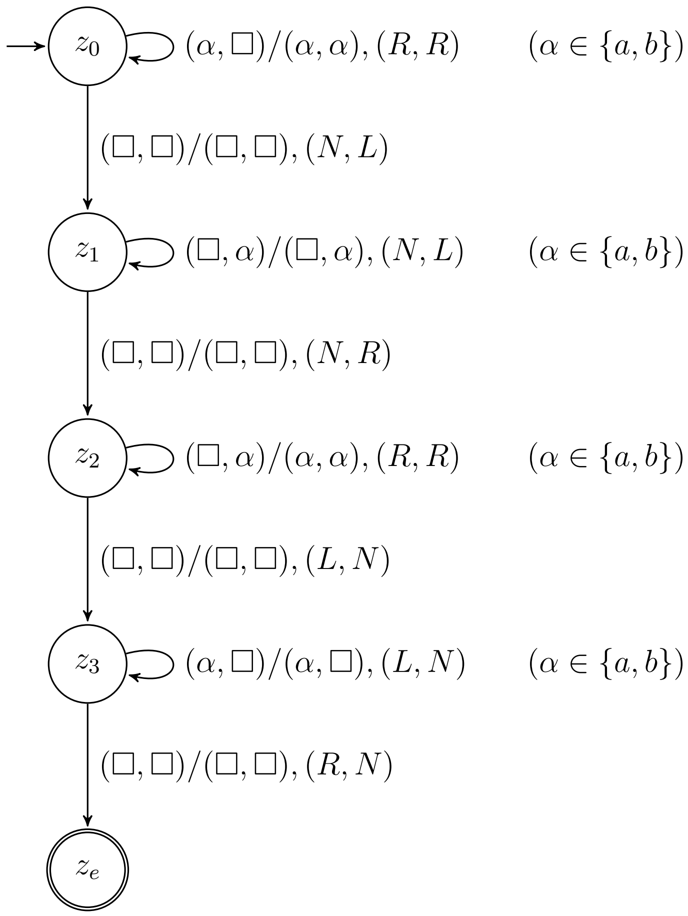
(c) Auf Eingabe des Wortes $\text{bin}(n)\#$ soll das Band der Turingmaschine am Ende der Berechnung das Wort $\text{bin}(4n+1)$ enthalten. Für $n > 0$ gilt $\text{bin}(4n+1) = \text{bin}(n)\,01$, so dass es genügt, das Zeichen $\#$ (statt $\Box$) durch das Wort 01 (statt 0 oder 1) zu ersetzen. Der Fall $n = 0$, der sich am ersten Zeichen erkennen lässt, muss gesondert betrachtet werden: Hier ersetzen wir die gesamte Eingabe $0\#$ durch das Wort 1.

Auf das abschließende Zurückbewegen des Lesekopf der Maschine an den Anfang der Ausgabe kann dabei verzichtet werden. Eine Turingmaschine mit der beschriebenen Arbeitsweise ist etwa wie folgt in Diagrammrepräsentation gegeben.



Aufgabe 3. Geben Sie eine Zwei-Band-Turingmaschine an, die bei Eingabe $w \in \{a, b\}^*$ zunächst das Wort ww auf das erste Band schreibt, dann den Lesekopf auf das erste Zeichen von ww bewegt und schließlich in einen Endzustand übergeht.

Lösung. Im Gegensatz zu einer Ein-Band-Konstruktion ist eine Zwei-Band-Turingmaschine deutlich weniger aufwendig. Die Idee ist, dass wir w vom ersten Band auf das zweite Band kopieren, während der Lesekopf auf dem ersten Band stehen bleibt, danach auf dem zweiten Band zum Anfang zurückfahren und schließlich die Kopie von w vom zweiten Band auf dem ersten Band hinter die Eingabe anhängen. Eine so arbeitende Turingmaschine ist, in Digrammrepräsentation, etwa wie folgt gegeben.



Aufgabe 4. Geben Sie Turingmaschinen M_i an, welche die Funktionen $\pi_i \colon \mathbb{N}^2 \to \mathbb{N}$ mit

$$\pi_i(n_1, n_2) = n_i \quad (i = 1, 2)$$

berechnen.

Lösung. Zu Beginn der Berechnung enthält das Band der Tuningmaschine M_i das Wort

$$\mathrm{bin}(n_1)\#\,\mathrm{bin}(n_2)\#$$

und nach Ablauf der Berechnung soll das Band nur noch das Wort $\mathrm{bin}(n_i)$ enthalten. Um das zu erreichen, durchlaufen wir das Band einmal von links nach rechts und löschen dabei alle Zeichen, die nicht Teil der Ausgaben sind. So arbeitende Turingmaschinen M_1 (links) und M_2 (rechts) sind, in Digrammrepräsentation, etwa wie folgt gegeben.