Prof. Javier Esparza
Kevin Kappelmann, Simon Roßkopf, Lukas Stevens

Technische Universität München
Lehrstuhl für Theoretische Informatik

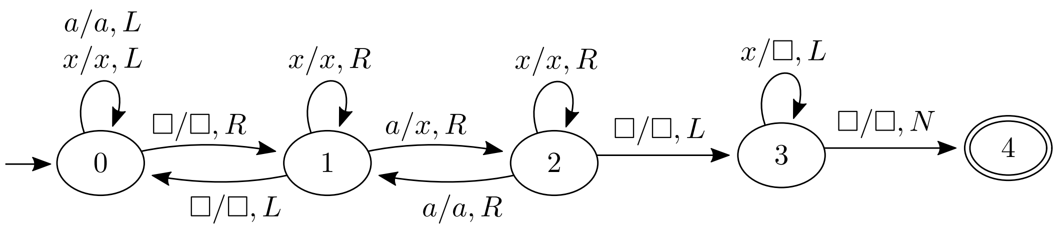
# Einführung in die Theoretische Informatik

Sommersemester 2023 – Hausaufgabenblatt 9

- Bitte beachten Sie, dass in dieser Vorlesung generell Antworten mit Begründung gefordert werden, solange die Aufgabe nicht explizit das Gegenteil sagt.
- Zum Bestehen diese Blattes müssen Sie 50% der Punkte erreichen.
- Update: Es wird die Aufgabe 3 (a,c,e,g) korrigiert.

**Aufgabe H9.1.** (*Jeu de mots*) 4 Punkte

Der Archäologe Jasper Vazarie hat eine antike Maschine ausgegraben und braucht Ihre Hilfe, um deren Bedeutung zu verstehen. Welche Sprache $L \subseteq \{a\}^*$ akzeptiert die folgende Maschine?




Begründen Sie Ihre Antwort und beschreiben Sie, wieso die TM die von ihnen genannte Sprache akzeptiert.

**Tipp:** Es gibt verschiedene Webseiten auf denen Turingmaschinen interaktiv konstruiert und simuliert werden können, z.B. `https://wimmers.github.io/turing-machine-viz/`oder `https://turingmachinesimulator.com/`.

*Lösungsskizze.* $\{a^{2^n} : n \in \mathbb{N}\}$. Die TM geht das Wort von links nach rechts durch, und ersetzt jedes zweite $a$ durch ein $x$. Falls es eine ungerade Anzahl an $a$ gibt, prüft sie, dass es genau ein $a$ ist (indem sie das erste $a$ durch ein $x$ ersetzt und schaut, dass danach nur $x$ übrigbleiben), und akzeptiert. Ansonsten wiederholt sich der Prozess. Damit wird die Anzahl an $a$ in jedem Schritt halbiert. Falls es am Anfang eine Zweierpotenz war, bleibt irgendwann ein einzelnes $a$ übrig und die Eingabe wird akzeptiert, ansonsten bleibt irgendwann eine größere ungerade Zahl an $a$ übrig, und die TM akzeptiert nicht. Auf Eingabe $\varepsilon$ hängt die TM, diese wird also (korrekterweise) nicht akzeptiert.

**Aufgabe H9.2.** (*...sehen deine Schuhe prima aus!*) 4 + 0 Punkte

Dr. Evilsparza lacht sich ins Fäustchen: endlich hat er eine allmächtige Unheilmaschine, den Schleifomaten, erfunden. Diese sind sogar mächtiger als Kellerautomaten, das habt ihr schließlich auf dem letzten Hausaufgabenblatt gezeigt... oder etwa nicht? Auf dem vorherigen Blatt habt ihr tatsächlich nur gezeigt, dass ein Schleifomat für eine nicht-kontextfreie Sprachen existiert. Aber können Sie auch wirklich alle kontextfreie Sprachen erkennen?

Ziel dieser Aufgabe ist es, Dr. Evilsparza endgültig zu überzeugen, dass Schleifomaten mächtiger als Kellerautomaten sind. Tatsächlich werdet ihr Dr. Evilsparza eine große Freude machen und sogar zeigen, dass sie mindestens so mächtig wie Turingmaschinen (und somit strikt mächtiger als Kellerautomaten) sind.

Wir definieren eine alternative Akzeptanzbedingung für Schleifomaten: Ein Schleifomat $A$ mit Endzuständen ist ein 7-Tupel $(Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$. Die ersten 6 Einträge bleiben unverändert, wie auf Hausaufgabenblatt 8 definiert. Der neue Eintrag $F \subseteq Q$ kennzeichnet die Endzustände des Schleifomats. Die mit Endzuständen akzeptierte Sprache eines Schleifomats ist definiert als

$$L_F(M) := \{w \in \Sigma^* \mid \exists q_f \in F, \gamma \in \Gamma^*.\ (q_0, w, Z_0) \to_M^* (q_f, \epsilon, \gamma)\},$$

wobei $\gamma$ die Ausgabe der Berechnung des Schleifomats ist. Wie auch bei Turingmaschinen nehmen wir an, dass $\delta(q_f, a)$ nicht definiert ist für $q_f \in F$ und $a \in \Gamma$.

(a) Beschreiben Sie, wie jede deterministische Turingmaschine durch einen Schleifomat mit Endzuständen simuliert werden kann. Dabei soll sowohl die Akzeptanz eines Wortes als auch die Ausgabe der Turingmaschine simuliert werden. Letzteres bedeutet, dass falls eine TM mit Band $\Box \cdots \Box w_1 \cdots w_n \Box \cdots \Box$ mit $w_1, \ldots, w_n \in \Gamma$ hält, die Schleife schlussendlich bei $\Box^k w_1 \cdots w_n \Box^m$ für beliebiges $k, m \in \mathbb{N}$ stehen soll. Ein ungültiger, finaler Schleifeninhalt wäre hingegen $w_2 \cdots w_n w_1$. Die finale Kopfposition der TM muss nicht ersichtlich sein.

   *Tipp:* Bilden sie eine Konfiguration $(v, q, w)$ der TM durch ein Wort $vqw\#$ in der Schleife ab.

(b) (Optional) Geben Sie nun ihre Konstruktion formal an (d.h. jede Komponente des 7-Tupels muss präzise definiert sein). Die Korrektheit der Konstruktion müssen Sie nicht beweisen.

*Lösungsskizze.*

(a) Sei $M = (Q, \Sigma, \Gamma, q_0, \delta, \Box, F)$ eine deterministische Turingmaschine, die wir durch einen Schleifomat $A$ simulieren wollen. Wir nehmen an, dass $Q \cap \Gamma = \emptyset$.

   Die Konstruktionsidee ist nun, dass wir eine Konfiguration $(v, q, w)$ der Turingmaschine als Wort $vqw\#$ in der Schleife ablegen. Für jede Transition der TM rotiert der Schleifomat den Zustand $q$ nach vorne und merkt sich dabei auch den letzten Buchstaben in $v$ (d.h. das Zeichen, das aktuell unter dem Kopf der TM steht). Dann führt der Schleifomat die dazugehörige Transition der TM aus und rotiert die Schleife wieder in die Anfangsposition. Der Schleifomat kann die Schleife rotieren, indem er ein Schleifenzeichen liest, und danach wieder ans Ende der Schleife anhängt. So lässt sich die Schleife flexibel manipulieren. Für jeden Schritt der Turingmaschine muss allerdings die Schleife einmal komplett durchlaufen werden. Wenn ein Endzustand erreicht wird, wird der Zustand und das #-Symbol von der Schleife gelöscht und der Schleifomat akzeptiert.

(b) Unser Schleifomat $A := (Q', \Sigma, \Gamma \cup Q \cup \{\#\}, \mathit{init}, \#, \delta', F)$ hat die Zustandsmenge

   $$Q' := \{\mathit{init}, \mathit{copy}, \mathit{start}\} \cup \{\mathit{find}_x \mid x \in \Gamma\} \cup \{\mathit{step}_x^q \mid x \in \Gamma, q \in Q\} \cup \{\mathit{ff}^q, \mathit{clean}^q \mid q \in Q\} \cup F$$

und die Übergänge

$$
\begin{array}{llll}
\delta'(start, \epsilon, x) & \to (find_x, \epsilon) & \forall x \in \Gamma & \text{Zum Kopf vorlaufen, dabei} \\
\delta'(find_x, \epsilon, y) & \to (find_y, x) & \forall x, y \in \Gamma & \text{jeweils letztes Zeichen merken.} \\
\delta'(find_x, \epsilon, q) & \to (step^q_x, \epsilon) & \forall x \in \Gamma, q \in Q & \text{Zustand auch merken.}
\end{array}
$$

$$
\delta'(step^q_x, \epsilon, y) \to \begin{cases} (ff^q, xq'z), & \text{falls } \delta(q,y) = (q', z, N) \\ (ff^q, xzq'), & \text{falls } \delta(q,y) = (q', z, R) \\ (ff^q, q'xz), & \text{falls } \delta(q,y) = (q', z, L) \end{cases} \quad \forall x, y \in \Gamma, q \in Q \quad \text{Schritt ausführen}
$$

$$
\begin{array}{llll}
\delta'(ff^q, \epsilon, x) & \to (ff^q, x) & \forall x \in \Gamma, \forall q \in Q & \text{Vorspulen bis zum \#\dots} \\
\delta'(ff^q, \epsilon, \#) & \to (start, \#) & \forall q \in Q \setminus F & \text{\dots und von vorne.} \\
\delta'(ff^q, \epsilon, \#) & \to (clean^q, \#) & \forall q \in F & \text{Zurück zum Start,} \\
 & & & \text{um den Zustand zu löschen.} \\
\delta'(clean^q, \epsilon, x) & \to (clean^q, x) & \forall x \in \Gamma, \forall q \in F & \text{Vorspulen zum Endzustand.} \\
\delta'(clean^q, \epsilon, q) & \to (clean^q, \epsilon) & \forall q \in F & \text{Zustand löschen} \\
\delta'(clean^q, \epsilon, \#) & \to (q, \epsilon) & \forall q \in F & \text{Lösche \# und akzeptiere} \\
\delta'(start, \epsilon, q) & \to (step^q_\Box, \epsilon) & \forall q \in Q & \text{Sonderfälle für Rand} \\
\delta'(step^q_x, \epsilon, \#) & \to (start, xq\Box\#) & \forall x \in \Gamma, q \in Q &
\end{array}
$$

Nun gilt (hier ohne Beweis):

$$
(start, \epsilon, vqw\#) \to_A^* (start, \epsilon, v'q'w'\#) \iff (v, q, w) \to_M^* (v', q', w')
$$

Um dafür zu sorgen, dass auch wirklich $L(A) = L(M)$ gilt, muss man den Schleifomat noch so erweitern, dass er am Anfang die initiale Konfiguration in die Schleife schreibt:

$$
\begin{array}{lll}
\delta'(init, x, \#) \to (copy, q_0 x\#) & \forall x \in \Sigma & \text{Initialzustand und erstes Zeichen schreiben} \\
\delta'(copy, \epsilon, x) \to (copy, x) & \forall x \in \Sigma \cup \{q_0\} & \text{Vorspulen bis zum \#} \\
\delta'(copy, x, \#) \to (copy, x\#) & \forall x \in \Sigma & \text{Nächstes Zeichen einlesen} \\
\delta'(copy, \epsilon, \#) \to (start, \#) & & \text{Einlesen beendet. Starte Simulation} \\
\delta'(init, \epsilon, \#) \to (start, q_0\#) & & \text{Sonderfall für leere Eingabe}
\end{array}
$$

**Aufgabe H9.3.** (*Wahr/Falsch*) 7*1 Punkte

Entscheiden Sie ob die folgenden Aussagen wahr oder falsch sind. Begründen Sie ihre Behauptung. Ein ausführlicher formaler Beweis ist nicht gefordert, aber eine umfassende Begründung.

(a) Die Klasse der "ruhelosen" Turingmaschinen die ihren Kopf in jedem Schritt bewegen müssen (d.h für alle Transition $\delta(q, a) = (q', b, D)$ muss $D \neq N$ sein) akzeptiert genau die Typ-0 Sprachen.

(b) Die Klasse der nichtdeterministischen Turing Maschinen die jede Zelle ihres Bandes höchstens zweimal ändern akzeptiert genau die Typ-0 Sprachen. Eine Transition $\delta(q, a) = (q', b, D)$ ändert eine Zelle wenn $a \neq b$.

(c) Die Klasse der nichtdeterministischen Turing Maschinen die jede Zelle ihres Bandes höchstens einmal ändern akzeptiert genau die kontextsensitiven Sprachen.

(d) Jede reguläre Sprache kann von einer Turingmaschine akzeptiert werden die keine Zelle auf ihrem Band ändert.

(e) Jede Sprache die von einer nicht-deterministischen Turingmaschine akzeptiert wird welche kein Leerzeichen überschreibt ist kontextfrei.

(f) Jede Sprache die von einer deterministischen Turingmaschine akzeptiert wird wird auch von einer nichtdeterministischen Turing Maschine akzeptiert.

(g) In einem unzerstörbaren, unknackbaren, uneinsehbaren Safe befindet sich ein Zettel auf dem eine jedem unbekannte natürliche Zahl $z$ steht. Die Funktion

$$f(n) = \begin{cases} 1 & \text{wenn } n = z \\ 0 & \text{sonst} \end{cases}$$

ist berechenbar.

*Lösungsskizze.*

(a) Wahr. Typ-0 Sprachen sind genau die Sprachen die von Turing Maschinen akzeptiert werden. Jede ruhelosen Turing Maschine ist offensichtlich eine Turing Maschine. Eine beliebige TM kann von einer ruhelosen Turing Maschine simuliert werden, indem man jede Transition die den Kopf nicht bewegt durch zwei Transitionen simuliert (Einen Schritt nach rechts, dann wieder einen nach links).

(b) Wahr. Jede TM die jede Zelle nur zweimal ändert ist eine TM. Noch zu zeigen jede TM $M$ kann von einer TM $M'$ die jede Zelle nur zweimal ändert simuliert werden:

Zu jedem Zeitpunkt steht nur endlich viel Information auf dem Band einer Turingmaschine. Die Idee ist es nun, für jeden Schritt der originalen Turingmaschine den aktuellen Bandinhalt auf einen noch freien Platz (z.B. rechts) zu kopieren und dabei einen Schritt der TM auszuführen. Dabei wird außerdem die Kopfposition der originalen TM auf dem Band mit einem neuen Markierungszeichen vermerkt. Die Sequenz an Konfigurationen kann dabei durch Trennzeichen abgegrenzt werden.

Initial fügt die Maschine das Kopfmarkierungszeichen ein. Die Maschine kopiert dann stets die aktuelle Konfiguration Zeichen für Zeichen nach rechts. Sobald ein Zeichen kopiert werden soll, wird es mit einer Markierung versehen. Falls das Zeichen nicht an der Kopfmarkierung anliegt, wird es einfach kopiert. Anderenfalls wird es entsprechend des Überganges der originalen Turingmaschine angepasst. Die Prozedur modifiziert jede Zelle somit höchstens zweimal: einmal, um ein Zeichen zum erstem mal hinzuschreiben und ein zweites mal, um das Zeichen im nächsten Schritt als kopiert zu markieren.

(c) Falsch. Auch diese Maschinen akzeptieren Typ-0 Sprachen. Jede TM die jede Zelle nur einmal ändert ist eine TM. Noch zu zeigen jede TM $M$ kann von einer TM $M'$ die jede Zelle nur einmal ändert simuliert werden:

Der Prozess ist analog zu dem vorherigen. Allerdings kann keine Markierung der Zeichen beim Kopieren in derselben Zelle passieren. Trick: Jede ursprüngliche Zelle wird durch zwei Zellen repräsentiert. Die erste Zelle enthält das ursprüngliche Zeichen und die zweite dient als Markierstelle während der Kopierprozedur. Der Input liegt nicht in diesem Format vor, allerdings darf dieser einmal überschrieben werden. Deshalb wird beim ersten

Kopiervorgang einfach die Markierung, wie schon vorher, über das Eingabezeichen gelegt (oder alternativ erstmal in das neue Format kopiert).

(d) Wahr. Für jede reguläre Sprache gibt es einen DFA der diese erkennt. Man kann jeden DFA simulieren indem man eine TM mit den gleichen Zuständen und einem zusätzlichen Endzustand konstruiert, welche die Zeichen der Eingabe nacheinander vom Band liest und ihren Zustand entsprechend den Transitionen des DFA ändert. Dabei wird der Kopf nur nach rechts bewegt und kein Zeichen überschrieben. Nur der neue Zustand ist Endzustand. Wenn die Maschine ein Leerzeichen liest ist das Wort vollständig gelesen. Befindet man in einem Zustand welcher im DFA Endzustand war wechselt man in den Endzustand der TM. Dieser zusätzliche Schritt ist notwendig da nach unserer Definition von Turing Maschinen keine Transitionen aus Endzuständen führen dürfen.

(e) Falsch.

Betrachte zum Beispiel $\{a^n b^n c^n | n \geq 0\}$ ist nicht kontextfrei, kann aber von einer Turing Maschine akzeptiert werden, welche keine Leerzeichen überschreibt.

Diese geht wie folgt vor:

(1) Wenn du ein a liest, ersetze es durch A und laufe nach rechts bis du ein b liest

(2) Ersetze dieses b durch B und laufe nach rechts bis du ein c siehst

(3) Ersetze dieses c durch C und laufe nach links bis du ein a siehst.

(4) Wiederhole diese Schritte bis alle a,b,c ersetzt sind, dann akzeptiere

(5) Wenn irgendwann ein unerwartetes Zeichen auftritt, lehne ab

(f) Wahr. Jede deterministische TM ist auch eine nichtdeterministische TM

(g) Wahr. Diese Funktion ist für jedes fixe $z$ berechenbar. Hierfür ist es irrelevant welches $z$ dies ist. $f$ ist berechenbar, da die Eingabe nur mit einer Konstante($z$) verglichen werden muss und je nach Ergebnis 0 oder 1 zurückgegeben wird.