Institut für Theoretische Informatik
Lehrstuhl Prof. Dr. D. Wagner

# Übungsblatt 3

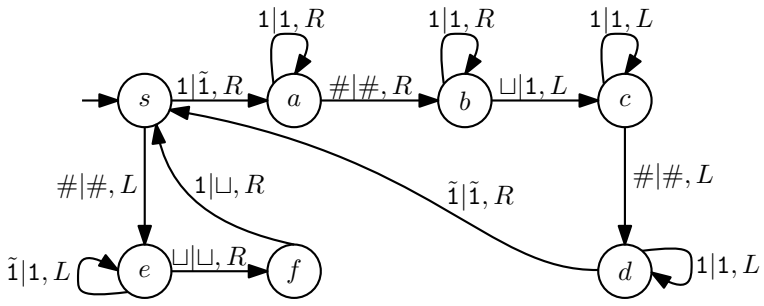
Vorlesung Theoretische Grundlagen der Informatik im WS 20/21

**Ausgabe:** 8. Dezember 2020
**Abgabe:** 22. Dezember 2020, 11:30 Uhr (digital im ILIAS)

## Aufgabe 1

(2 + 1 + 2 + 2 = 7 Punkte)

Die Turing-Maschine $\mathcal{M} = (Q, \Sigma = \{\mathtt{1}, \#\}, \Gamma = \{\mathtt{1}, \tilde{\mathtt{1}}, \#, \sqcup\}, \delta, s, \emptyset)$ wird durch den folgenden Zustandsgraphen beschrieben.




(a) Simulieren Sie $\mathcal{M}$ auf der Eingabe $\mathtt{111}\#$, bis Sie zum dritten Mal in den Zustand $s$ gelangen. Geben Sie alle Konfigurationen an, die dabei durchlaufen werden.

(b) Simulieren Sie $\mathcal{M}$ mit der Konfiguration $\tilde{\mathtt{1}}\tilde{\mathtt{1}}\tilde{\mathtt{1}}(s)\#\mathtt{111}$, bis Sie erneut in den Zustand $s$ gelangen.

(c) Was ist die Aufgabe der Zustände $\{s, a, b, c, d\}$ und $\{s, e, f\}$?

(d) Welche Funktion berechnet $\mathcal{M}$?

**Lösung:**

(a)
- $(s)\mathtt{111}\#$
- $\tilde{\mathtt{1}}(a)\mathtt{11}\#$
- $\tilde{\mathtt{1}}\mathtt{1}(a)\mathtt{1}\#$
- $\tilde{\mathtt{1}}\mathtt{11}(a)\#$
- $\tilde{\mathtt{1}}\mathtt{11}\#(b)\sqcup$
- $\tilde{\mathtt{1}}\mathtt{11}(c)\#\mathtt{1}$
- $\tilde{\mathtt{1}}\mathtt{1}(d)\mathtt{1}\#\mathtt{1}$
- $\tilde{\mathtt{1}}(d)\mathtt{11}\#\mathtt{1}$

- $(d)\tilde{1}11\#1$
- $\tilde{1}(s)11\#1$
- $\tilde{1}\tilde{1}(a)1\#1$
- $\tilde{1}\tilde{1}1(a)\#1$
- $\tilde{1}\tilde{1}1\#(b)1$
- $\tilde{1}\tilde{1}1\#1(b)\sqcup$
- $\tilde{1}\tilde{1}1\#(c)11$
- $\tilde{1}\tilde{1}1(c)\#11$
- $\tilde{1}\tilde{1}(d)1\#11$
- $\tilde{1}(d)\tilde{1}1\#11$
- $\tilde{1}\tilde{1}(s)1\#11$

(b)
- $\tilde{1}\tilde{1}\tilde{1}(s)\#111$
- $\tilde{1}\tilde{1}(e)\tilde{1}\#111$
- $\tilde{1}(e)\tilde{1}1\#111$
- $(e)\tilde{1}11\#111$
- $(e)\sqcup 111\#111$
- $(f)111\#111$
- $(s)11\#111$

(c) $\{s, a, b, c, d\}$ kopiert die unmarkierten 1-en vor dem Trennsymbol # ans Ende vom Band. $\{s, e, f\}$ dekrementiert die unär kodierte Zahl vor dem Trennsymbol # um 1 und entfernt die Markierungen.

(d) Bei der Eingabe $1^n\#, n \in \mathbb{N}_0$ erzeugt $\mathcal{M}$ das Wort $\#1^m$ mit $m = \Sigma_{i=1}^{n} i$.

## Aufgabe 2

(1 + 1 + 1 = 3 Punkte)

Betrachten Sie die folgenden Entscheidungsprobleme:

(a) Lassen sich die Zeichen eines Worts $w$ über dem Alphabet $\Sigma = \{0, 1, \ldots, 9\}$ zu einem Wort $w'$ umsortieren, sodass für alle aufeinanderfolgenden Zeichen $x, y$ in $w'$ gilt, dass $|x - y| = 1$?

(b) BinPacking: Gegeben eine Menge $M \subset \mathbb{Q}$ und ein $k \in \mathbb{N}$, existiert eine Partition $M_1, M_2, \ldots, M_k$ von $M$ mit $\sum_{m \in M_i} m \leq 1$ für $1 \leq i \leq k$?

(c) Coloring: Gegeben ein Graph $G = (V, E)$ und ein $k \in \mathbb{N}$, existiert eine Funktion $f : V \to \{1, 2, \ldots, k\}$ mit $f(u) \neq f(v)$ für alle $\{u, v\} \in E$?

Geben Sie für jedes dieser Probleme eine Ja-Instanz und eine Nein-Instanz an. Beschreiben Sie außerdem, wie eine Orakel-TM arbeitet, die das Problem in polynomieller Zeit löst. Geben Sie dazu insbesondere an:

(i) wie die Instanzen kodiert werden,

(ii) wie die Orakel-TM den Lösungsvorschlag des Orakelmoduls interpretiert und

(iii) was der deterministische Teil der Orakel-TM tun muss, um den Lösungsvorschlag zu überprüfen.

**Lösung:**

(a)
- Ja-Instanz: 0211
- Nein-Instanz: 0102
- Kodierung: Identitätsfunktion
- Interpretiere Lösungsvorschlag des Orakelmoduls als $w'$.
- Der deterministische Teil der Orakel-TM muss überprüfen, ob jedes Zeichen aus $\Sigma$ gleich oft in $w$ und $w'$ vorkommt. Außerdem muss überprüft werden, ob für alle aufeinanderfolgenden Zeichen $x, y$ in $w'$ gilt, dass $|x - y| = 1$.

(b)
- Ja-Instanz: $M = \{1/2, 2/3, 1/4\}$, $k = 2$.
- Nein-Instanz: $M = \{1/2, 2/3, 3/4\}$, $k = 2$.
- Kodierung: Sei $M = \{a_1/b_1, a_2/b_2, \ldots, a_n/b_n\}$.
- Kodiere die Instanz über dem Alphabet $\Sigma = \{0, 1, \#\}$ als

$$\text{bin}(k)\#\,\text{bin}(n)\#\,\text{bin}(a_1)\#\,\text{bin}(b_1)\#\,\text{bin}(a_2)\#\,\text{bin}(b_2)\#\ldots\#\,\text{bin}(a_n)\#\,\text{bin}(b_n)$$

- Interpretiere Lösungsvorschlag des Orakelmoduls als Partition $M_1, M_2, \ldots, M_k$ (geeignet kodiert).
- Der deterministische Teil der Orakel-TM muss überprüfen, ob tatsächlich $\sum_{m \in M_i} m \leq 1$ für alle $1 \leq i \leq n$ gilt.

(c)
- Ja-Instanz: $G = (\emptyset, \emptyset), k = 1$.
- Nein-Instanz: $G = (\{u, v\}, \{\{u, v\}\}), k = 1$.
- Kodierung der Instanz: Sei $V = \{1, 2, \ldots, n\}$ und $E = \{\{u_1, v_1\}, \{u_2, v_2\}, \ldots, \{u_m, v_m\}\}$.
- Kodiere die Instanz über dem Alphabet $\Sigma = \{0, 1, \#\}$ als

$$\text{bin}(n)\#\,\text{bin}(m)\#\,\text{bin}(u_1)\#\,\text{bin}(v_1)\#\,\text{bin}(u_2)\#\,\text{bin}(v_2)\#\ldots\#\,\text{bin}(u_m)\#\,\text{bin}(v_m)$$

- Interpretiere den Lösungsvorschlag des Orakelmoduls als Funktion $f$ kodiert als

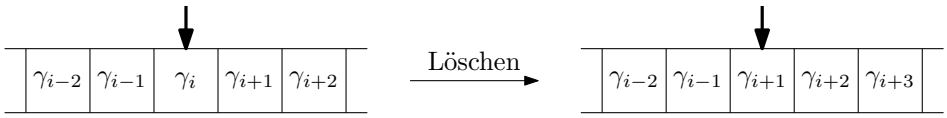
$$\text{bin}(f(v_1))\#\,\text{bin}(f(v_2))\#\ldots\#\,\text{bin}(f(v_n)).$$

- Der deterministische Teil der Orakel-TM muss überprüfen, ob tatsächlich $f(u_i) \neq f(v_i)$ für $1 \leq i \leq m$ gilt.

## Aufgabe 3

(1 + 2 + 3 = 6 Punkte)

Eine *löschende Turing-Maschine* (LTM) ist eine deterministische Turing-Maschine, der zusätzlich zu den Kopfbewegungen *links*, *keine Bewegung* und *rechts* die Aktion *löschen* zur Verfügung steht. Beim Löschen wird das Feld, auf dem der Kopf der Turing-Maschine steht, aus dem Band gelöscht und der Kopf bewegt sich auf das Feld, das zuvor der rechte Nachbar des nun gelöschten Feldes war:




Sei $L = \{w\#w^R \mid w \in \{0,1\}^*\}$.

(a) Welche Laufzeit braucht eine deterministische Turing-Maschine mit einem Band, um $L$ zu entscheiden? Wählen Sie aus den folgenden Möglichkeiten die schärfste untere Schranke für die Worst-Case-Laufzeit aus.

- ☐ $\Omega(\log n)$
- ☐ $\Omega(n)$
- ☐ $\Omega(n^2)$
- ☐ $\Omega(2^n)$

Begründen Sie Ihre Entscheidung kurz (ohne formalen Beweis).

(b) Beschreiben Sie, wie man $L$ mit einer LTM in Linearzeit entscheiden kann. Belegen Sie die Laufzeit.

(c) Sind Turing-Maschinen und löschende Turing-Maschinen gleich mächtig? Beweisen Sie.

**Lösung:**

(a) $\Omega(n^2)$. Eine Turing-Maschine „muss“ je zwei Zeichen von $w$ und $w^R$ vergleichen. Dabei müssen insgesamt Distanzen von $\Omega(\Sigma_{i=1}^{n} i) = \Omega(n^2)$ auf dem Band zurückgelegt werden.

(b) Suche die Mitte des Worts, die mit # markiert ist, oder lehne die Eingabe ab, falls keine solche Markierung existiert. Dies ist in linearer Laufzeit möglich. Lösche dann dieses Symbol. Vergleiche nun das aktuelle Symbol mit dem linken Nachbarsymbol. Gleichen sich die Symbole nicht, so lehne die Eingabe ab. Gleichen sich die Symbole, so lösche beide und wiederhole diesen Schritt, bis die komplette Eingabe gelöscht wurde. Dies ist wiederum in linearer Laufzeit möglich.

(c) Ja. Eine LTM kann trivialerweise eine TM simulieren. Umgekehrt kann eine TM $\mathcal{M}$ auch eine LTM $\mathcal{L}$ simulieren. Löscht $\mathcal{L}$ ein Zeichen, so muss $\mathcal{M}$ den Teil des Bandes rechts vom aktuellen Bandsymbol um eine Position nach links verschieben. Das Ende des Bandes muss dazu z.B. durch ein zusätzliches Bandsymbol geeignet markiert werden.

## Aufgabe 4

(1 + 1 + 1 + 1 = 4 Punkte)

Zeigen oder widerlegen Sie:

(a) Das Komplement des Halteproblems ist semi-entscheidbar.

(b) Das Komplement der Diagonalsprache ist semi-entscheidbar.

(c) Seien $L_1$ und $L_2$ semi-entscheidbare Sprachen. Dann ist auch $L_1 \cup L_2$ semi-entscheidbar.

(d) Seien $L_1$ und $L_2$ semi-entscheidbare Sprachen. Dann ist auch $L_1 \setminus L_2$ semi-entscheidbar.

**Lösung:**

(a) Falsch. Aus der Vorlesung wissen wir, dass das Halteproblem semi-entscheidbar ist. Angenommen, das Komplement des Halteproblems wäre ebenfalls semi-entscheidbar. Dann könnte man die beiden entsprechenden Turing-Maschinen „pseudoparallel“ laufen lassen, also abwechselnd jeweils einen Schritt der entsprechenden Turing-Maschine simulieren. Da die beiden Turing-Maschinen komplementäre Sprachen akzeptieren, hält mindestens eine von beiden nach endlicher Zeit. Damit wäre das Halteproblem entscheidbar. Widerspruch.

(b) Richtig. Das Komplement der Diagonalsprache ist $L_d^c = \{w_i \mid M_i \text{ akzeptiert } w_i\}$. Ist $w_i \in L_d^c$, kann dies durch Simulation von $M_i$ auf der Eingabe $w_i$ in endlicher Zeit festgestellt werden. Damit ist $L_d^c$ semi-entscheidbar.

(c) Richtig. Seien $L_1$ und $L_2$ semi-entscheidbare Sprachen und $M_1$ und $M_2$ Turing-Maschinen, die $L_1$ bzw. $L_2$ akzeptieren. Konstruiere eine Turing-Maschine $M$, die $L_1 \cup L_2$ akzeptiert. Dazu werden bei Eingabe von $w$ die Maschinen $M_1$ und $M_2$ „pseudoparallel“ jeweils mit Eingabe $w$ simuliert. Das funktioniert, indem abwechselnd jeweils ein Schritt der beiden Simulationen ausgeführt wird. Akzeptiert eine der beiden Simulationen die Eingabe, akzeptiert auch $M$. Gilt $w \in L_1$ oder $w \in L_2$, so akzeptiert mindestens eine der beiden Simulationen nach endlicher Zeit, sodass auch $M$ nach endlicher Zeit akzeptiert. Damit akzeptiert $M$ die Sprache $L_1 \cup L_2$.

Hier ist es wichtig, dass die Simulationen nicht einfach hintereinander ausgeführt werden. Dann wäre es nämlich möglich, dass die erste Simulation nicht terminiert, wodurch die zweite Simulation nie feststellen kann, dass die Eingabe zur gesuchten Sprache gehört.

(d) Falsch. Sei $L_1 = \Sigma^*$ und $L_2 = \mathcal{H}$ (das Halteproblem). Beide Sprachen sind semi-entscheidbar. Es gilt $L_1 \setminus L_2 = \Sigma^* \cap \mathcal{H}^c = \mathcal{H}^c$. Aus (a) wissen wir aber, dass das Komplement des Halteproblems nicht semi-entscheidbar ist.

## Aufgabe 5

(2 + 3 = 5 Punkte)

Eine Turing-Maschine $M$ zählt eine unendliche Sprache $L$ auf, wenn $M$ niemals stoppt und eine Liste $w_1, w_2, \ldots$ genau der Wörter aus $L$ ausgibt. Dabei ignoriert $M$ die Eingabe und die Wörter der ausgegebenen Liste sind eindeutig voneinander getrennt. Für die Reihenfolge der Wörter in der Liste muss gelten, dass jedes Wort aus $L$ nach endlich vielen Schritten ausgegeben wird. Eine unendliche Sprache $L$ ist aufzählbar, falls eine Turing-Maschine existiert, die $L$ aufzählt.

(a) Zeigen Sie, dass $L$ genau dann entscheidbar ist, wenn $L$ in kanonischer Reihenfolge[1] aufzählbar ist.

(b) Zeigen Sie, dass $L$ genau dann semi-entscheidbar ist, wenn $L$ aufzählbar ist. Erklären Sie auch, warum sich hier im Gegensatz zu Aufgabenteil (a) nicht fordern lässt, dass die Reihenfolge der Aufzählung kanonisch ist.

**Lösung:**

[1] Siehe Vorlesung 6, Folie 20. Die kanonische Reihenfolge sortiert Wörter nach aufsteigender Länge und Wörter der gleichen Länge lexikographisch.

(a) Sei $L$ eine aufzählbare Sprache zusammen mit einer Turing-Maschine $M$, die $L$ in kanonischer Reihenfolge aufzählt. Konstruiere eine Turing-Maschine $M'$, die für jedes $w$ entscheidet, ob es zu $L$ gehört. Dazu simuliert $M'$ die Turing-Maschine $M$ solange, bis das erste Wort $w' \geq w$ aufgezählt wird. Da $M$ die Sprache $L$ aufzählt, geschieht dies nach endlicher Zeit. Gilt $w' = w$, so folgt $w \in L$, andernfalls gilt $w \notin L$. Die Turing-Maschine $M'$ entscheidet also $L$.

Sei umgekehrt $L$ eine entscheidbare Sprache mit einer entscheidenden Turing-Maschine $M$. Konstruiere eine Turing-Maschine $M'$ mit separatem Arbeits- und Ausgabeband, die die Sprache $L$ in kanonischer Reihenfolge aufzählt. Dazu schreibt $M'$ in kanonischer Reihenfolge alle Wörter in $\Sigma^*$ auf das Arbeitsband. Für jedes Wort $w$ wird dann $M$ mit $w$ als Eingabe simuliert. Da $L$ entscheidbar ist, stoppt $M$ in jedem Fall. Wird $w$ von $M$ akzeptiert, schreibe $w$ auf das Ausgabeband.

(b) Sei $L$ eine aufzählbare Sprache zusammen mit einer aufzählenden Turing-Maschine $M$. Konstruiere eine Turing-Maschine $M'$, die $L$ semi-entscheidet. Sei $w$ eine Eingabe für $M'$. Simuliere die Turing-Maschine $M$ solange, bis $w$ ausgegeben wird, und akzeptiere dann. Falls $w \in L$ ist wird $M'$ das Wort $w$ nach endlicher Zeit akzeptieren. Damit ist $L$ eine semi-entscheidbare Sprache.

Sei umgekehrt $L$ eine semi-entscheidbare Sprache mit einer semi-entscheidenden Turing-Maschine $M$. Konstruiere eine Turing-Maschine $M'$, die die Sprache $L$ aufzählt. Dazu simuliert $M'$ die Turing-Maschine $M$ „pseudoparallel“ für alle Wörter $w_1, w_2, \ldots$ in kanonischer Reihenfolge. Das funktioniert folgendermaßen. In Schritt 1 simuliert $M'$ einen Schritt von $M$ auf der Eingabe $w_1$. In Schritt 2 simuliert $M'$ einen (weiteren) Schritt von $M$ auf den Eingaben $w_1, w_2$. In Schritt 3 simuliert $M'$ einen (weiteren) Schritt von $M$ auf den Eingaben $w_1, w_2, w_3$, und so weiter. Sobald $M'$ eine Eingabe $w$ akzeptiert, schreibt $M'$ das Wort $w$ auf das Ausgabeband. Da $L$ semi-entscheidbar ist, wird jedes Wort $w \in L$ irgendwann auf das Ausgabeband geschrieben. Also zählt $M'$ die Sprache $L$ auf. Man bemerke, dass es hier wichtig ist, die Simulationen „pseudoparallel“ ablaufen zu lassen, damit eine nicht-terminierende Simulation andere, terminierende Simulationen nicht aufhält.

Betrachte zwei Wörter $w' > w$. Die Laufzeit von $M$ für die Eingaben $w'$ und $w$ kann sehr unterschiedlich sein, weshalb es vorkommen kann, dass $w'$ vor $w$ ausgegeben wird. Daher ist die Reihenfolge der Ausgabe nicht unbedingt kanonisch.

## Aufgabe 6 (3 Punkte)

Zeigen Sie, dass die Sprache $L = \{\langle M \rangle \mid L(M) = \emptyset\}$ unentscheidbar ist.

**Lösung:**
Beweis durch Widerspruch. Angenommen, $L$ wäre durch eine Turing-Maschine $\mathcal{R}$ entscheidbar. Konstruiere daraus einen Entscheider $\mathcal{S}$ für die universelle Sprache. Die Idee ist, dass $\mathcal{S}$ zu einer Turingmaschine $\mathcal{M}$ und Eingabe $w$ eine neue Turingmaschine $\mathcal{M}'$ konstruiert, sodass $L(\mathcal{M}') = \emptyset$ genau dann, wenn $\mathcal{M}$ die Eingabe $w$ nicht akzeptiert. Die TM $\mathcal{M}'$ verwirft ihre eigene Eingabe $x$, simuliert $\mathcal{M}$ auf $w$ und gibt dann dieselbe Antwort wie $\mathcal{M}$. Es gilt also $L(\mathcal{M}') = \emptyset$ genau dann, wenn $\mathcal{M}$ die Eingabe $w$ nicht akzeptiert. Die TM $\mathcal{S}$ simuliert nun $\mathcal{R}$ auf $\langle \mathcal{M}' \rangle$. Falls $\mathcal{R}$ akzeptiert, so lehnt $\mathcal{S}$ ab, ansonsten akzeptiert $\mathcal{S}$.

## Aufgabe 7 (1 + 2 + 1 = 4 Punkte)

In der Übung wurde das Äquivalenzproblem für Turing-Maschinen vorgestellt:

$$L_{\text{äq}} = \{w\#v \in \{0,1,\#\}^* \mid L(T_w) = L(T_v)\}$$

In dieser Aufgabe sollen Sie zeigen, dass weder $L_{\text{äq}}$ noch $L_{\text{äq}}^c$ semi-entscheidbar ist. Gehen Sie dazu wie folgt vor:

(a) Zeigen Sie, dass das Komplement der universellen Sprache $L_u^c$ nicht semi-entscheidbar ist.

(b) Zeigen Sie, dass $L_{\text{äq}}$ nicht semi-entscheidbar ist. Nehmen Sie dazu an, es gäbe eine Turing-Maschine $M_{\text{äq}}$, die $L_{\text{äq}}$ akzeptiert. Konstruieren Sie daraus eine Turing-Maschine $M_u^c$, die $L_u^c$ akzeptiert.

(c) Zeigen Sie mit der gleichen Herangehensweise wie in Aufgabenteil (b), dass $L_{\text{äq}}^c$ nicht semi-entscheidbar ist.

**Lösung:**

(a) In der Vorlesung wurde gezeigt, dass $L_u$ nicht entscheidbar, aber semi-entscheidbar ist. Außerdem wurde in der Übung gezeigt, dass eine Sprache $L$ genau dann entscheidbar ist, wenn $L$ und $L^c$ semi-entscheidbar sind. Wäre $L_u^c$ also entscheidbar, wäre $L_u$ entscheidbar, was zum Widerspruch führt.

(b) Wir konstruieren eine Turing-Maschine $M_u^c$, die $L_u^c$ akzeptiert, wie folgt:

- Die Eingabe von $M_u^c$ sind eine Turing-Maschine $M$ und ein Wort $w$. $M_u^c$ muss genau dann akzeptieren, falls $w$ nicht von $M$ akzeptiert wird.
- Konstruiere eine Turing-Maschine $M_1$, die ihre Eingabe ignoriert und stattdessen $M$ auf $w$ simuliert.
- Konstruiere eine Turing-Maschine $M_2$, die alle Eingaben ablehnt.
- Simuliere $M_{\text{äq}}$ auf der Eingabe $\langle M_1\rangle\#\langle M_2\rangle$. Falls $M_{\text{äq}}$ akzeptiert, akzeptiere. Sonst lehne ab.

Es gilt:

$$L(M_1) = \begin{cases} \{0,1\}^* & \text{falls } w \text{ von } M \text{ akzeptiert wird} \\ \emptyset & \text{sonst} \end{cases}$$

Außerdem gilt $L(M_2) = \emptyset$. Es gilt also $L(M_1) = L(M_2)$ genau dann, wenn $w$ von $M$ nicht akzeptiert wird. Genau in diesem Fall akzeptiert $M_u^c$, d.h. $M_u^c$ akzeptiert $L_u^c$.

(c) Die Konstruktion ist gleich zu Aufgabenteil (b), außer dass $M_2$ nun eine Turing-Maschine ist, die alle Eingaben akzeptiert, und für die Simulation $M_{\text{äq}}^c$ statt $M_{\text{äq}}$ verwendet wird. Nun gilt also $L(M_2) = \{0,1\}^*$ und somit gilt $L(M_1) \neq L(M_2)$ genau dann, wenn $w$ von $M$ nicht akzeptiert wird. Genau in diesem Fall akzeptiert $M_u^c$, d.h. $M_u^c$ akzeptiert $L_u^c$.