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

Übungsblatt 3

Vorlesung Theoretische Grundlagen der Informatik im WS 21/22

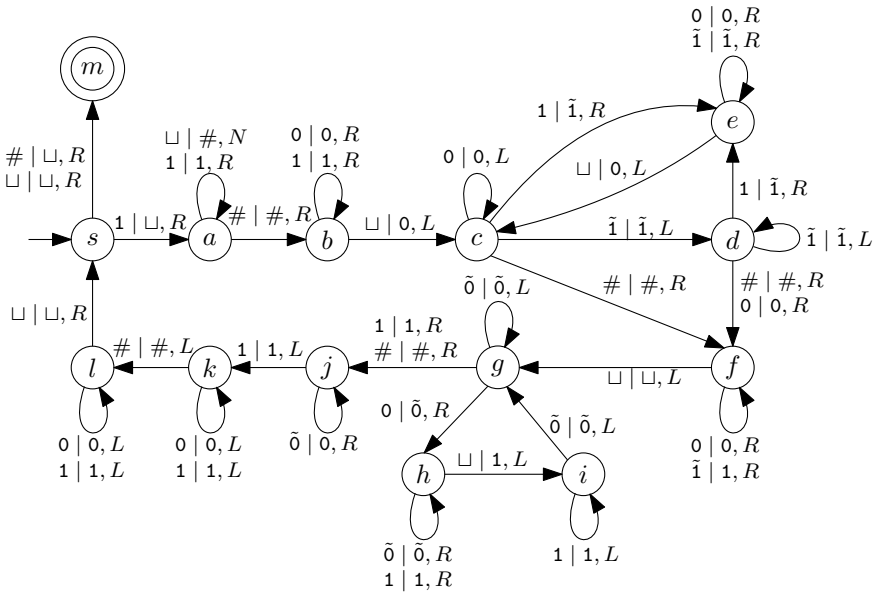
Ausgabe: 19. November 2021
Abgabe: 3. Dezember 2021 (digital im ILIAS)

Bitte bearbeiten Sie die Aufgaben **handschriftlich** und laden Sie eine eingescannte PDF-Version im Übungsmodul Ihrer ILIAS-Tutoriumsgruppe hoch! Beschriften Sie Ihren handschriftlichen Aufschrieb gut sichtbar mit Name und Matrikelnummer. Nicht handschriftliche oder unbeschriftete Abgaben werden nicht akzeptiert!

Aufgabe 1

(2 + 1 + 2 + 1 = 6 Punkte)

Die Turing-Maschine $\mathcal{M} = (Q, \Sigma = \{1\}, \Gamma = \{0, \tilde{0}, 1, \tilde{1}, \#, \sqcup\}, \delta, s, \{m\})$ wird durch den folgenden Zustandsgraphen beschrieben.



(a) Simulieren Sie die Turing-Maschine $\mathcal{M}$ auf der Eingabe 11, bis Sie zum zweiten Mal in den Zustand s gelangen. Geben Sie alle Konfigurationen an, die dabei auftreten.

(b) Welche Ausgabe produziert die Turing-Maschine $\mathcal{M}$ auf der Eingabe 11?

(c) Was ist die Aufgabe der Zustandsmengen $\{s, a, b, c, e, d, f\}$, $\{g, h, i\}$ und $\{j, k, l\}$?

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

Lösung:

(a)
- $(s)\mathsf{11}$
- $(a)\mathsf{1}$
- $\mathsf{1}(a)\sqcup$
- $\mathsf{1}(a)\#$
- $\mathsf{1}\#(b)\sqcup$
- $\mathsf{1}(c)\#\mathsf{0}$
- $\mathsf{1}\#(f)\mathsf{0}$
- $\mathsf{1}\#\mathsf{0}(f)\sqcup$
- $\mathsf{1}\#(g)\mathsf{0}$
- $\mathsf{1}\#\tilde{\mathsf{0}}(h)$
- $\mathsf{1}\#(i)\tilde{\mathsf{0}}\mathsf{1}$
- $\mathsf{1}(g)\#\tilde{\mathsf{0}}\mathsf{1}$
- $\mathsf{1}\#(j)\tilde{\mathsf{0}}\mathsf{1}$
- $\mathsf{1}\#\mathsf{0}(j)\mathsf{1}$
- $\mathsf{1}\#(k)\mathsf{01}$
- $\mathsf{1}(k)\#\mathsf{01}$
- $(l)\mathsf{1}\#\mathsf{01}$
- $(l)\sqcup\mathsf{1}\#\mathsf{01}$
- $(s)\mathsf{1}\#\mathsf{01}$

(b) $\mathsf{010011}$

(c) Wenn die Turing-Maschine wieder im Zustand s ankommt, ist sie in einer Konfiguration $(s)\mathsf{1}^p\#\mathsf{010011000111}\ldots\mathsf{0}^q\mathsf{1}^q$ für $p, q \in \mathbb{N}_0$. Falls $p > 0$, dann wird die linkeste Eins gelöscht und die Zustände $\{s, a, b, c, e, d, f\}$ dienen dazu, dass ganz rechts noch $q+1$ Nullen angehängt werden. Dafür wird zuerst eine Null geschrieben und danach werden die Einser aus $\mathsf{1}^q$ nach und nach markiert und für jede davon wird noch eine Null angehängt. Als Ergebnis ist der Bandinhalt:

$$\mathsf{1}^{p-1}\#\mathsf{010}^2\mathsf{1}^2\mathsf{0}^3\mathsf{1}^3\ldots\mathsf{0}^q\mathsf{1}^q\mathsf{0}^{q+1}.$$

Danach dienen die Zustände $\{g, h, i\}$ dazu, dass man noch $q+1$ Einser ganz rechts anhängt: dafür werden die rechtesten $q+1$ Nullen nacheinander markiert und für jede Markierung wird eine Eins geschrieben.

Schließlich werden die Zustände $\{j, k, l\}$ benutzt, um die Markierungen zu löschen und den Schreiblesekopf wieder nach ganz links zu bringen.

(d) Der oben beschriebene Vorgang wird für Eingabe $\mathsf{1}^k$ genau k Mal wiederholt und schließlich wird # gelöscht. Die Turing-Maschine $\mathcal{M}$ berechnet somit die Funktion $f : \Sigma^* \to \Gamma^*$ so, dass für jedes $k \in \mathbb{N}_0$:

$$f(\mathsf{1}^k) = \mathsf{010}^2\mathsf{1}^2\mathsf{0}^3\mathsf{1}^3\ldots\mathsf{0}^k\mathsf{1}^k.$$

Aufgabe 2 (2 + 2 = 4 Punkte)

Betrachten Sie die folgenden Entscheidungsprobleme:

(a) Gegeben eine endliche Menge $M \subset \mathbb{N}$ von natürlichen Zahlen, existieren Zahlen $a, b, c \in M$ mit $a + b^2 = c^3$?

(b) LONGEST PATH: Gegeben ein Graph $G = (V, E)$ und ein $k \in \mathbb{N}$, existiert ein Pfad in G mit k Knoten? (Ein Pfad mit k Knoten ist eine Folge von paarweise unterschiedlichen Knoten $v_1, \ldots, v_k \in V$, sodass $v_i v_{i+1} \in E$ für alle $i \in \{1, \ldots, k-1\}$.)

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. Begründen sie kurz die polynomielle Laufzeit.

Lösung:

(a)
- Ja-Instanz: $M = \{11, 4, 3\}$
- Nein-Instanz: $M = \{1\}$
- Sei $M = \{p_1, p_2, \ldots, p_n\}$. Kodiere die Instanz über dem Alphabet $\Sigma = \{0, 1, \#\}$ als

$$\text{bin}(p_1)\#\,\text{bin}(p_2)\#\ldots\#\,\text{bin}(p_{n-1})\#\,\text{bin}(p_n)$$

- Interpretiere den Lösungsvorschlag des Orakelmoduls als

$$\text{bin}(a)\#\,\text{bin}(b)\#\,\text{bin}(c) \tag{1}$$

- Der deterministische Teil der Orakel-TM muss Folgendes überprüfen:
  1. Der Vorschlag hat die Form (1).
  2. Die Zahlen a, b, c kommen in M vor.
  3. Es gilt $a + b^2 = c^3$.

  Das geht in Linearzeit, insbesondere also auch in polynomieller Zeit.

(b)
- Ja-Instanz: $G = (\{1, 2, 3, 4, 5\}, \{\{1, 2\}, \{2, 3\}, \{4, 5\}\}, k = 3$.

- Nein-Instanz: $G = (\{1, 2, 3, 4, 5\}, \{\{1, 2\}, \{2, 3\}, \{4, 5\}\}, k = 4$.
- 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:

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

- Interpretiere den Lösungsvorschlag als Folge von Knoten $z_1, z_2, \ldots, z_k$ kodiert als:

$$\text{bin}(z_1)\#\,\text{bin}(z_2)\#\ldots\#\,\text{bin}(z_k) \tag{2}$$

- Der deterministische Teil der Orakel-TM muss Folgendes überprüfen:
  1. Der Vorschlag hat die Form (2).

2. Es gilt $1 \leq z_i \leq n$ für jedes $i \in \{1, \ldots, k\}$.
3. Die Knoten sind paarweise unterschiedlich: Für alle $i < j \in \{1, \ldots, k\}$ gilt $z_i \neq z_j$.
4. Die Knoten bilden einen Pfad: Für alle $i \in \{1, \ldots, k-1\}$: $z_i z_{j+1} \in E$ existiert ein $j \in \{1, \ldots, m\}$ mit $\{z_i, z_{i+1}\} = \{u_j, v_j\}$.

Da die Knoten paarweise unterschiedlich sein müssen, enthält eine Lösung höchstens n Knoten. Die Existenz der Kanten kann naiv in $\mathcal{O}(n^3)$ überprüft werden (jedes Mal über alle Kanten des Graphen iterieren). Damit ist auch die Überprüfung insgesamt in polynomieller Laufzeit.

Aufgabe 3

(3 + 2 = 5 Punkte)

Wir zeigen in dieser Aufgabe, dass die Sprache

$$L = \{\langle \mathcal{M} \rangle \mid L(\mathcal{M}) \text{ ist unentscheidbar}\}$$

unentscheidbar ist.

(a) Für eine Eingabe $w\#v$, konstruieren Sie eine Turing-Maschine $\mathcal{M}^{wv}$, sodass gilt:

$$L(\mathcal{M}^{wv}) \text{ ist unentscheidbar} \iff T_w \text{ akzeptiert } v.$$

(b) Zeigen Sie, dass L unentscheidbar ist. Nehmen Sie dazu an, dass es eine Turing-Maschine $\mathcal{M}_L$ gibt, die L entscheidet, und verwenden Sie $\mathcal{M}_L$, um die universelle Sprache zu entscheiden.

Sie dürfen die Aussage der Teilaufgabe (a) verwenden, auch wenn Sie diese nicht gezeigt haben.

Lösung:

(a) Für eine Eingabe $w\#v$ konstruieren wir eine Turing-Maschine $\mathcal{M}^{wv}$, die genau dann das Komplement der Diagonalsprache akzeptiert, wenn T_w das Wort v akzeptiert. Wenn v nicht von T_w akzeptiert wird, soll $\mathcal{M}^{wv}$ nur die leere Sprache akzeptieren.

Dabei arbeitet $\mathcal{M}^{wv}$ für gegebene w und v wie folgt: Für jede Eingabe x ignoriert $\mathcal{M}^{wv}$ die Eingabe zunächst und simuliert erst T_w auf v.

Wenn v nicht von T_w akzeptiert wird, übernimmt $\mathcal{M}^{wv}$ das Ergebnis (also hält nicht oder lehnt ab). In diesem Fall akzeptiert $\mathcal{M}^{wv}$ keine Eingabe x, also gilt $L(\mathcal{M}^{wv}) = \emptyset$.

Wenn T_w v akzeptiert, insbesondere also auch hält, simuliert $\mathcal{M}^{wv}$ danach T_x auf der Eingabe x und akzeptiert genau dann, wenn x von T_x akzeptiert wird. In diesem Fall akzeptiert $\mathcal{M}^{wv}$ eine Eingabe x genau dann, wenn $x \in L(T_x)$, also gilt $L(\mathcal{M}^{wv}) = L_d^c$.

Da die leere Sprache entscheidbar ist, das Komplement der Diagonalsprache aber nicht, hat die Turing-Maschine $\mathcal{M}^{wv}$ für gegebene w und v die geforderte Eigenschaft.

(b) Wir nehmen an, dass L entscheidbar ist. Dann existiert eine Turing-Maschine $\mathcal{M}_L$, die L entscheidet. Wir konstruieren mit Hilfe von $\mathcal{M}_L$ eine Turing-Maschine $\mathcal{M}_U$, die die universelle Sprache entscheidet.

Sei $w\#v$ die Eingabe von $\mathcal{M}_U$. Zunächst konstruiert $\mathcal{M}_U$ die in Teilaufgabe (a) beschriebene Turing-Maschine $\mathcal{M}^{wv}$. Dann wird $\mathcal{M}_L$ auf der Eingabe $\langle \mathcal{M}^{wv} \rangle$ simuliert und das Akzeptanzverhalten von $\mathcal{M}_L$ übernommen. Es gilt also:

$$w\#v \in L(\mathcal{M}_U) \iff \langle \mathcal{M}^{wv} \rangle \in L(\mathcal{M}_L) \iff L(\mathcal{M}^{wv}) \text{ ist unentscheidbar} \overset{(a)}{\iff} v \in L(T_w)$$

Da $\mathcal{M}_L$ die Sprache L entscheidet, insbesondere also auch immer hält, hält auch $\mathcal{M}_U$ immer. Insgesamt akzeptiert $\mathcal{M}_U$ eine Eingabe $w\#v$, wenn $w\#v$ in der universellen Sprache ist und lehnt sonst ab, entscheidet also damit die universelle Sprache. Da aber die universelle Sprache unentscheidbar ist, muss auch L unentscheidbar sein.

Aufgabe 4 (2 Punkte)

Betrachten Sie die Sprache

$$L = \{\langle \mathcal{M} \rangle \mid L(\mathcal{M}) = \emptyset\}.$$

Zeigen Sie, dass L^c semi-entscheidbar ist.

Lösung:
Es gilt:

$$L^c = \{\langle \mathcal{M} \rangle \mid L(\mathcal{M}) \neq \emptyset\} = \{\langle \mathcal{M} \rangle \mid \mathcal{M} \text{ akzeptiert mindestens ein Wort}\}.$$

Für eine gegebene Turing-Maschine M kann man nicht einfach alle möglichen Eingaben in Σ^* nacheinander ausprobieren, da es möglich ist, dass M auf einer Eingabe gar nicht hält (und damit die Eingaben danach nicht mehr ausprobiert werden). Stattdesssen ist im Folgenden die Idee, dass man alle möglichen Eingaben pseudoparallel (also quasi gleichzeitig) abarbeitet.

Wir konstruieren eine Turing-Maschine $\mathcal{M}_L$, die L akzeptiert. Die Turing-Maschine $\mathcal{M}_L$ habe ein zweidimensionales Band (aus der Vorlesung ist bekannt, dass so eine Turing-Maschine durch eine normale 1-Band Turing-Maschine simuliert werden kann). Für ein $i \in \mathbb{N}$ nennen wir alle Felder des Bandes der Form (i, j) *die i-te Zeile*. Betrachte die kanonische Reihenfolge der Wörter aus $\{0,1\}^*$. Zur Erinnerung: In dieser Reihenfolge kommt ein Wort v vor einem Wort w, falls

- $|v| < |w|$ oder
- $|v| = |w|$ und v ist lexikographisch kleiner als w.

Also sieht diese Reihenfolge folgendermaßen aus:

$$w_1 = \varepsilon, w_2 = \mathsf{0}, w_3 = \mathsf{1}, w_4 = \mathsf{00}, w_5 = \mathsf{01}, w_6 = \mathsf{10}, w_7 = \mathsf{11}, w_8 = \mathsf{000}, w_9 = \mathsf{001}, w_{10} = \mathsf{010}, \ldots$$

Jetzt können wir die Turing-Maschine $\mathcal{M}_L$ definieren. Sei $\langle \mathcal{M} \rangle$ eine Eingabe. Die Turing-Maschine $\mathcal{M}_L$ arbeitet in Iterationen. In der i-ten Iteration schreibt $\mathcal{M}_L$ zuerst in der i-ten Zeile das Wort w_i auf das Band und simuliert danach jeweils den nächsten Schritt von $\langle \mathcal{M} \rangle$ auf jedem der Wörter $w_1, w_2, \ldots, w_i$. Falls eine von diesen Simulationen akzeptiert, dann akzeptiert auch $\mathcal{M}_L$. Wir behaupten, dass

$$L(\mathcal{M}_L) = L^c.$$

Sei zuerst $\langle \mathcal{M} \rangle \in L(\mathcal{M}_L)$ beliebig. Also wurde $\langle \mathcal{M} \rangle$ von der Turing-Maschine $\mathcal{M}_L$ akzeptiert. Nach Konstruktion gibt es ein Wort w_j so, dass die Simulation von $\langle \mathcal{M} \rangle$ auf der Eingabe w_j akzeptiert hat, also gilt

$$w_j \in L(\mathcal{M})$$

und somit

$$L(\mathcal{M}) \neq \emptyset.$$

Deswegen gilt

$$\langle \mathcal{M} \rangle \in L^c.$$

Da $\langle \mathcal{M} \rangle \in L(\mathcal{M}_L)$ beliebig gewählt wurde, gilt:

$$L(\mathcal{M}_L) \subseteq L^c.$$

Sei jetzt $\langle \mathcal{M} \rangle \in L^c$ beliebig, dann gilt:

$$L(\mathcal{M}) \neq \emptyset.$$

Also existiert ein Wort $w \in \Sigma^*$ mit:

$$w \in L(\mathcal{M}).$$

Sei j so, dass $w = w_j$ und sei i die Anzahl der Schritte, nach der $\mathcal{M}$ das Wort w akzeptiert. Dann wird die Turing-Maschine $\mathcal{M}_L$ das Wort w in der j-ten Iteration auf das Band schreiben (wenn die Turing-Maschine $\mathcal{M}_L$ nicht schon akzeptiert hat) und dann wird in der $j+i$-ten Iteration die Simulation von $\mathcal{M}$ das Wort w akzeptieren (wenn die Turing-Maschine $\mathcal{M}_L$ nicht schon akzeptiert hat). Also akzeptiert die Turing-Maschine $\mathcal{M}_L$ die Eingabe $\langle \mathcal{M} \rangle$. Und somit:

$$\langle \mathcal{M} \rangle \in L(\mathcal{M}_L).$$

Und da $\langle \mathcal{M} \rangle \in L^c$ beliebig war, gilt:

$$L^c \subseteq L(\mathcal{M}_L).$$

Insgesamt gilt:

$$L^c = L(\mathcal{M}_L).$$

Somit akzeptiert $\mathcal{M}_L$ genau die Sprache L^c und diese Sprache ist semi-entscheidbar.

Beachte: Mit diesem Verfahren zeigen wir nur die Semi-Entscheidbarkeit von L^c. Für eine Eingabe $\langle \mathcal{M} \rangle \notin L^c$ wird kein w_i gefunden, das von $\mathcal{M}$ akzeptiert wird und somit hält $\mathcal{M}_L$ auf der Eingabe $\langle \mathcal{M} \rangle$ nicht an. Tatsächlich sind die Sprachen L und L^c nicht entscheidbar.

Aufgabe 5

(1 + 2 + 1 + 1 + 1 + 2 = 8 Punkte)

Zeigen oder widerlegen Sie:

(a) Sei $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, s, F)$ eine deterministische Turing-Maschine, die eine Sprache $L \subseteq \Sigma^*$ akzeptiert. Dann akzeptiert die Turing-Maschine $\mathcal{M}' = (Q, \Sigma, \Gamma, \delta, s, Q \setminus F)$ genau die Sprache L^c.

(b) Sei L eine entscheidbare Sprache und L_1, L_2 semi-entscheidbare Sprachen mit:

$$L_1 \cap L_2 = \emptyset, L_1 \cup L_2 = L.$$

Dann ist L_1 auch entscheidbar.

(c) Sei L eine entscheidbare Sprache und L_1, L_2 semi-entscheidbare Sprachen mit:

$$L_1 \cup L_2 = L.$$

Dann ist L_1 auch entscheidbar.

(d) Sei L_1 eine entscheidbare Sprache und $L_1 \cap L_2$ eine entscheidbare Sprache. Dann ist auch L_2 entscheidbar.

(e) Jede unentscheidbare Sprache enthält eine entscheidbare Teilmenge.

(f) Für jede unentscheidbare Sprache gibt es eine echte Obermenge, die ebenfalls unentscheidbar ist.

Lösung:

(a) Die Aussage ist falsch. Sei $L = L_u$ die universelle Sprache. Aus der Vorlesung ist bekannt, dass L semi-entscheidbar ist. Sei $\mathcal{M}$ die Turing-Maschine, die L akzeptiert. Jetzt nehmen wir an, dass die Aussage stimmt. Dann akzeptiert die Turing-Maschine $\mathcal{M}'$ die Sprache L^c. Laut der Vorlesung gilt: Wenn L und L^c semi-entscheidbar sind, dann ist $L = L_u$ entscheidbar. Das ist aber ein Widerspruch zu Nicht-Entscheidbarkeit der universellen Sprache. Also stimmt die Aussage nicht.

(b) Die Aussage stimmt. Sei $\mathcal{M}$ eine TM, die L entscheidet, und sei $\mathcal{M}_i$ (für $i \in \{1, 2\}$) eine TM, die L_i akzeptiert. Wir konstruieren eine 3-Band-TM $\mathcal{M}'$, die L_1 entscheidet. Laut Vorlesung kann so eine TM durch eine normale TM simuliert werden. Sei w eine Eingabe. Sie wird von $\mathcal{M}'$ auf das zweite und auf das dritte Band kopiert. Zuerst simuliert $\mathcal{M}'$ die Ausführung von $\mathcal{M}$ auf w auf dem ersten Band.

Wenn die Simulation von $\mathcal{M}$ ablehnt, dann lehnt auch $\mathcal{M}'$ ab. Danach wiederholt $\mathcal{M}'$ das Folgende: Führe einen Schritt von $\mathcal{M}_1$ auf dem zweiten Band und einen Schritt von $\mathcal{M}_2$ auf dem dritten Band aus. Falls die Simulation von $\mathcal{M}_1$ akzeptiert, dann akzeptiert auch $\mathcal{M}'$. Falls die Simulation von $\mathcal{M}_2$ akzeptiert, dann lehnt $\mathcal{M}'$ ab.

Betrachte eine beliebige Eingabe $w \in \Sigma^*$. Beobachte, dass

$$\Sigma^* = L \cup L^c = (L_1 \cup L_2) \cup L^c = L_1 \cup L_2 \cup L^c.$$

Nach Voraussetzung sind L_1 und L_2 disjunkt. Weiterhin gilt

$$L_1, L_2 \subseteq L.$$

Also gilt

$$L_1 \cap L^c = L_2 \cap L^c = \emptyset.$$

Somit sind L_1, L_2 und L^c paarweise disjunkt. Also liegt w entweder in L_1, in L_2, oder in L^c.

Sei zuerst $w \in L^c$ (also $w \notin L_1$). Dann lehnt die Simulation von $\mathcal{M}$ ab, also lehnt auch $\mathcal{M}'$ ab.

Sei jetzt $w \in L_1$ (also gilt $w \in L, w \notin L_2$). Also akzeptiert zuerst die Simulation von $\mathcal{M}$. Danach akzeptiert Simulation von $\mathcal{M}_1$, aber nicht von $\mathcal{M}_2$. Also akzeptiert $\mathcal{M}'$.

Sei schließlich $w \in L_2$ (also gilt $w \in L, w \notin L_1$). Also akzeptiert zuerst die Simulation von $\mathcal{M}$. Danach akzeptiert die Simulation von $\mathcal{M}_2$, aber nicht von $\mathcal{M}_1$. Also lehnt $\mathcal{M}'$ ab.

Somit hält $\mathcal{M}'$ auf jeder Eingabe und akzeptiert genau die Wörter aus L_1. Also ist L_1 entscheidbar.

(c) Die Aussage stimmt nicht. Seien $L = \Sigma^*$, $L_1 = L_u$ (die universelle Sprache), $L_2 = \Sigma^*$. Dann ist L entscheidbar (betrachte eine TM, die die Eingabe ignoriert und direkt in einen akzeptierenden Zustand übergeht). Somit ist $L_2 = L$ auch semi-entscheidbar und $L_1 = L_u$ ist semi-entscheidbar laut Vorlesung. Es gilt:

$$L_1 \cup L_2 = L_u \cup \Sigma^* = \Sigma^* = L.$$

Allerdings ist $L_1 = L_u$ nicht entscheidbar laut Vorlesung.

(d) Die Aussage stimmt nicht. Seien $L_1 = \emptyset$, $L_2 = L_d$ (die Diagonalsprache) und $L = \emptyset$. Dann ist $L_1 = L_1 \cap L_2 = \emptyset$ entscheidbar (betrachte eine TM, die die Eingabe ignoriert und direkt in einen ablehnenden Zustand übergeht), aber $L_2 = L_d$ ist nicht entscheidbar laut Vorlesung.

(e) Die Aussage stimmt. Die leere Sprache $L = \emptyset$ ist Teilmenge jeder Sprache und entscheidbar.

(f) Die Aussage stimmt. Sei L eine unentscheidbare Sprache. Nach (c) gilt $L \neq \Sigma^*$. Also gibt es ein Wort $w' \notin L$. Sei

$$L' = L \cup \{w'\}.$$

Dann ist L' eine echte Obermenge von L. Angenommen, es gibt eine Turing-Maschine $\mathcal{M}'$, die L' entscheidet. Daraus konstruieren wir eine Turing-Maschine $\mathcal{M}$, die L entscheidet, wie folgt. Bei Eingabe w überprüft zuerst $\mathcal{M}$, ob $w = w'$ gilt. Falls ja, lehnt $\mathcal{M}$ ab. Sonst simuliert $\mathcal{M}$ die Turing-Maschine $\mathcal{M}'$ auf w und akzeptiert genau dann, wenn diese Simulation akzeptiert.

Da $\mathcal{M}'$ auf jeder Eingabe hält, hält $\mathcal{M}$ auch auf jeder Eingabe. Weiterhin:

- Falls $w = w'$, dann gilt $w \notin L$ nach der Wahl von w' und $\mathcal{M}$ lehnt ab.
- Falls $w \neq w'$ und $w \notin L$, dann gilt $w \notin L'$ und die Simulation von $\mathcal{M}'$ lehnt w ab, da $\mathcal{M}'$ genau L' entscheidet, also lehnt auch $\mathcal{M}$ ab.
- Falls $w \in L$, dann gilt insbesondere $w \neq w'$ und $w \in L'$. Also akzeptiert die Simulation von $\mathcal{M}'$ und somit akzeptiert auch $\mathcal{M}$.

Somit akzeptiert $\mathcal{M}$ genau die Sprache L. Da $\mathcal{M}$ auf jeder Eingabe hält, entscheidet sie L. Widerspruch zu Unentscheidbarkeit von L. Also war die Annahme falsch und L' ist eine unentscheidbare echte Obermenge von L.

Aufgabe 6 (2 + 3 + 2 = 7 Punkte)

Wir erweitern das Modell einer Turing-Maschine, indem wir einen weiteren Kopf hinzufügen. Die Köpfe haben eine gemeinsame Zustandsmenge Q und gemeinsame Eingabe- und Bandalphabete Σ und Γ. Beide Köpfe arbeiten auf dem selben Band, wobei abwechselnd je ein Schritt der Übergangsfunktionen für Kopf 1 und Kopf 2 ausgeführt wird. Auch wenn Kopf 2 an der Reihe ist, ist bekannt, was Kopf 1 gerade liest, und umgekehrt. Die Übergangsfunktion hat also die Form

$$\delta \colon Q \times \Gamma \times \Gamma \times \{1, 2\} \to Q \times \Gamma \times \{L, N, R\},$$

wobei der vierte Parameter angibt, welcher Kopf an der Reihe ist. Die Übergangsfunktion wird also abwechselnd mit 1 bzw. 2 als viertes Argument aufgerufen. Wir nennen eine solche Turing-Maschine eine 2-*Kopf-Turing-Maschine*.

(a) Sei $\Sigma = \{a, b, \#\}$. Beschreiben Sie eine 2-Kopf-Turing-Maschine, die die Spiegelsprache $\{w\#w^R \mid w \in \{a, b\}^*\}$, wobei w^R das Spiegelwort von w bezeichne, in Linearzeit erkennt. Begründen Sie kurz die Laufzeit.

(b) Zeigen Sie, dass 2-Kopf-Turing-Maschinen und klassische Turing-Maschinen (d.h. 1 Kopf, 1 Band, wie in Vorlesung definiert) gleich mächtig sind. Simulieren Sie hierzu eine klassische Turing-Maschine mit einer 2-Kopf-Turing-Maschine und umgekehrt.

(c) Wir entfernen nun den vierten Parameter aus der Übergangsfunktion, sodass sich beide Köpfe bei gleichem Zustand und gleichen gelesenen Symbolen gleich verhalten. Die Übergangsfunktion hat jetzt also die Form

$$\delta' \colon Q \times \Gamma \times \Gamma \to Q \times \Gamma \times \{L, N, R\}.$$

Wir nennen eine solche Turing-Maschine eine *Zwillingskopf-Turing-Maschine*.

Zeigen Sie, dass 2-Kopf-Turing-Maschinen und Zwillingskopf-Turing-Maschinen gleich mächtig sind. Simulieren Sie hierzu eine Zwillingskopf-Turing-Maschine mit einer 2-Kopf-Turing-Maschine und umgekehrt.

Lösung:

(a) Kopf 2 läuft bis zum letzten Zeichen der Eingabe. Kopf 1 bleibt währenddessen stehen und ändert das Band nicht. Falls kein Trennzeichen existiert, ablehnen. Danach abwechselnd: Kopf 1 vergleicht die beiden Symbole unter den Köpfen. Bei Gleichheit, einen Schritt nach rechts; bei Ungleichheit ablehnen. Kopf 2 geht einen Schritt nach links. Wenn beide Köpfe das Trennzeichen lesen und auf dem gleichen Feld stehen, wird die Eingabe akzeptiert. Wenn sie beim Lesen des Trennzeichens nicht auf dem gleichen Feld stehen, oder wenn sie auf dem gleichen Feld stehen ohne das Trennzeichen zu lesen, wird abgelehnt. Bei Eingabelänge n läuft Kopf 2 am Anfang n Schritte nach rechts. Danach läuft Kopf 1 nur nach rechts und Kopf 2 nur nach links, bis sie sich in der Mitte treffen. Insgesamt benötigt die 2-Kopf-Turing-Maschine also höchstens $2n$ Schritte.

(b) Eine 2-Kopf-Turing-Maschine kann eine klassische Turing-Maschine simulieren, indem der zweite Kopf nicht verwendet wird. Umgekehrt kann eine klassische Turing-Maschine eine 2-Kopf-Turing-Maschine simulieren, indem sie zwischen den beiden Kopfpositionen hin- und herläuft. Dazu müssen die Kopfpositionen gespeichert werden. Dies kann durch ein größeres Bandalphabet $\Gamma' = \Gamma \times \{-, \alpha_1, \alpha_2, \alpha_{12}\}$ realisiert werden. Das Symbol $(\gamma, -)$ wird statt γ verwendet, wenn auf diesem Feld kein Kopf steht, die Symbole (γ, α_1) bzw. (γ, α_2), wenn der entsprechende Kopf auf dem Feld steht, und (γ, α_{12}), wenn beide Köpfe auf dem Feld stehen. Außerdem merken wir uns über die Zustandskontrolle (Verdoppelung der Anzahl an Zuständen), ob Kopf 1 links von Kopf 2 ist oder andersherum. Dadurch wissen wir beim Wechseln der Köpfe, in welche Richtung wir laufen müssen. Beim Laufen zum anderen Kopf merken wir uns mit Hilfe von Zuständen, welches Zeichen wir zuletzt gelesen haben.

(c) Eine 2-Kopf-Turing-Maschine kann eine Zwillings-Turing-Maschine simulieren, indem für beide Köpfe die Übergangsfunktion der Zwillings-Turing-Maschine gewählt wird. Umgekehrt kann eine Zwillings-Turing-Maschine $\mathcal{M}_Z$ eine 2-Kopf-Turing-Maschine $\mathcal{M}_2$ simulieren, indem sie sich über die Zustände merkt, welcher Kopf gerade an der Reihe ist. Sei hierzu Q_2 die Menge der Zustände von $\mathcal{M}_2$. Die Zustandsmenge Q_Z von $\mathcal{M}_Z$ wird definiert als $Q_Z = Q_2 \times \{1, 2\}$. Der Startzustand von $\mathcal{M}_Z$ ist $(s, 1)$, wobei s den Startzustand von $\mathcal{M}_2$ bezeichnet. Die Übergangsfunktion wird übernommen, wobei es für jeden Übergang mit Kopf i in $\mathcal{M}_2$ von Zustand q in q', in $\mathcal{M}_Z$ den Übergang $(q, i) \mapsto (q', 3 - i)$ gibt.

Aufgabe 7 (0 Punkte)

Diese Aufgabe war eine Tutoriumsaufgabe und nicht Teil des Übungsblatts!

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 8 (0 Punkte)

Diese Aufgabe war eine Tutoriumsaufgabe und nicht Teil des Übungsblatts!

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:

(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 Σ^* 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.

[1] Siehe Vorlesung 6, Folie 22. Die kanonische Reihenfolge sortiert Wörter nach aufsteigender Länge und Wörter der gleichen Länge lexikographisch.