Grundzüge der Theoretischen Informatik, WS 21/22: Musterlösung zum 9. Übungsblatt

Julian Dörfler

Wichtig: Sie dürfen von diesem Blatt Aufgaben Aufgaben im Wert von bis zu 24 Punkten[1] bearbeiten. Die Aufgaben auf diesem Blatt sind nach ungefährer Schwierigkeit sortiert.
Hinweis: Die Teilaufgaben aus A9.1-A9.5(b) stellen typische Klausurteilaufgaben dar.

Aufgabe A9.1 (Quiz) *(4 Bonuspunkte)*
Entscheiden Sie, welche der folgenden Aussagen wahr, und welche falsch sind und begründen Sie Ihre Antworten. Die Begründung sollte nicht länger als ein oder zwei Sätze sein.

(a) Jede totale Funktion ist `WHILE`-berechenbar.

(b) Jede Menge, die keine nicht-triviale Indexmenge ist, ist entscheidbar.

(c) Das Komplement einer endlichen Menge ist entscheidbar.

(d) Für jedes Paar $A, B \subseteq \mathbb{N}$ gilt $A \leq B$ oder $B \leq A$.

(e) Jede totale `WHILE`-berechenbare Funktion ist `FOR`-berechenbar.

(f) Es gibt eine Menge $A \subseteq \mathbb{N}$, so dass $A \leq H_0$ und $H_0 \leq \overline{A}$.

(g) Die einzige endliche Indexmenge ist die leere Menge.

(h) Es gibt eine Turingmaschine M die, gegeben die Binärdarstellung einer Gödelisierung g eines `WHILE`-Programms mit einer 1 auf dem Band terminiert, wenn g auf g hält und ansonsten mit einer 0 auf dem Band terminiert.

Lösung A9.1 (Quiz)

(a) Falsch, ein Gegenbeispiel ist z.B. die charakteristische Funktion des Halteproblems.

(b) Falsch, aus dem Vorlesungsskript wissen, wir, dass das Halteproblem keine Indexmenge ist. Entscheidbar ist es trotzdem nicht.

(c) Wahr, die Aussage ist äquivalent dazu, dass eine endliche Menge entscheidbar ist, was wir schon gezeigt hatten.

(d) Falsch, seien $A = H_0$ und $B = \overline{H_0}$. Dann folgt aus der Aussage, dass $\overline{H_0} \leq H_0$ (wir erinnern uns, dass $A \leq B \Leftrightarrow \overline{A} \leq \overline{B}$). Damit folgt aber, dass $\overline{H_0} \in \mathsf{RE}$, was ein Widerspruch ist.

[1] Bitte zerteilen Sie hierbei keine Aufgaben in ihre Teilaufgaben

(e) Falsch, zB $U_{\texttt{FOR}}$.

(f) Falsch: Aus $A \leq H_0$ folgt, dass $A \in \mathsf{RE}$. Aus $H_0 \leq \overline{A}$ folgt $\overline{H_0} \leq A$, also $A \notin \mathsf{RE}$, was ein Widerspruch ist.

(g) Wahr: Eine endliche Menge ist entscheidbar und nach dem Satz von Rice sind nichttriviale Indexmengen nicht entscheidbar. Die einzigen trivialen Indexmengen sind die leere Menge und die Menge aller Gödelisierungen, wobei letztere nicht endlich ist.

(h) Falsch: Diese Turingmaschine berechnet die charakteristische Funktion des speziellen Halteproblems, und da `WHILE`-berechenbarkeit und Turingberechenbarkeit äquivalent sind, würde folgen dass $H_0 \in \mathsf{REC}$ ist, was ein Widerspruch ist.

Aufgabe A9.2 (Nichtreguläre Sprachen) *(4 Bonuspunkte)*
Zeigen Sie, dass folgende Sprachen nicht regulär sind:

(a) $A = \{0^{2^n} \mid n \in \mathbb{N}\}$.

(b) $B = \{(0^\star 1)^{n^2} \mid n \in \mathbb{N}\}$.

(c) $C = \{0^i 1^j 2^k \mid i \cdot j \neq k\}$.

(d) $D = \{x \in \{(,)\}^\star \mid x \text{ ist ein korrekter Klammerausdruck}\}$.

Ein Klammerausdruck $x = x_1 \dots x_n$ ist korrekt, wenn

(i) für alle $\ell \in \{1, \dots, n\}$ der String $x_1 \dots x_\ell$ mindestens so viele öffnende wie schließende Klammern enthält und

(ii) x gleich viele öffnende wie schließende Klammern enthält.

Lösung A9.2 (Nichtreguläre Sprachen) (a) Sei $n \in \mathbb{N} \setminus \{0\}$ beliebig. Wir nehmen an $n \geq 2$, ansonsten mache ab nun mit $n = 2$ weiter. Wähle $uvw = 0^{2^n} \in A$ mit $u = \varepsilon, v = 0^n$ und $w = 0^{2^n - n}$. Sei nun $xyz = v$ mit $|y| = \ell \geq 1$. Wir wählen $i = 2$ und erhalten das Wort $uxy^izw = 0^{2^n + \ell}$. Nun gilt aber $2^n < 2^n + \ell \leq 2^n + n < 2^n + 2^n = 2^{n+1}$, $2^n + \ell$ ist also keine Zweierpotenz. Es gilt also $0^{2^n+\ell} \notin A$.

A lässt sich also nicht pumpen und ist damit nicht regulär.

(b) Sei h ein Homomorphismus mit $h(0) = \varepsilon$ und $h(1) = 1$. Dann ist $h(B) = \{1^{n^2} \mid n \in \mathbb{N}\} =: B'$. Wäre B nun regulär, so wäre B' ebenfalls regulär, da REG unter Homomorphismen abgeschlossen ist. Nach Aufgabe P4.1 ist B' aber nicht regulär und somit also $B \notin \mathsf{REG}$.

(c) Die Strings 01^a und 01^b für $a \neq b$ sind nicht Myhill-Nerode-äquivalent: $01^a2^a \notin C$, aber $01^b2^a \in C$. Da $a, b \in \mathbb{N}$ beliebig waren, besitzt C also unendlich viele Myhill-Nerode-Äquivalenzklassen und ist damit nicht regulär.

Alternativer Beweis: Wir verwenden die Abgeschlossenheit von REG unter Mengendifferenz und Schnitt. Wir betrachten

$$C' = (\{0,1,2\}^\star \setminus C) \cap L(0^\star 1^\star 2^\star) = \{0^i 1^j 2^k \mid i \cdot j = k\}$$

Falls C also regulär ist, so muss C' ebenfalls regulär sein.

Nun zeigen wir, dass C' nicht regulär ist mithilfe des Pumping-Lemmas:
Sei $n \in \mathbb{N} \setminus \{0\}$ beliebig. Wähle $uvw = 01^n2^n$, mit $u = 01^n, v = 2^n$ und $w = \varepsilon$. Sei nun $xyz = v$ mit $|y| = \ell \geq 1$. Wir wählen $i = 0$ und erhalten $uxy^izw = 01^n2^{n-\ell}$. Wegen $\ell > 0$ ist aber $n - \ell < 1 \cdot n$, also $uxy^0zw \notin C'$.

C' lässt sich also nicht pumpen und ist somit nicht regulär. Damit ist also auch C nicht regulär.

(d) Die Strings $(^a$ und $(^b$ für $a \neq b$ sind nicht Myhill-Nerode-äquivalent: $(^a)^a \in D$, aber $(^b)^a \notin D$. Da $a, b \in \mathbb{N}$ beliebig waren, besitzt D also unendlich viele Myhill-Nerode-Äquivalenzklassen und ist damit nicht regulär.

Aufgabe A9.3 (Best of Computability Theory 1) *(8 Bonuspunkte)*
Entscheiden Sie für jede der folgenden Mengen ob diese (1) eine Indexmenge, (2) entscheidbar, (3) rekursiv aufzählbar und/oder (4) co-rekursiv aufzählbar sind. Beweisen Sie Ihre Antworten.

(a) $L_1 = \{i \in \mathbb{N} \mid \varphi_i \text{ hat mind. eine Nullstelle}\}$

(b) $L_2 = \{i \in \mathbb{N} \mid \varphi_i(i) = (i+1)^2\}$

Lösung A9.3 (Best of Computability Theory 1)

(a) L_1 ist eine Indexmenge (1): Wenn $i \in L_1$ und $j \in \mathbb{N}$ mit $\varphi_j = \varphi_i$, dann hat φ_j auch mindestens eine Nullstelle. Ferner ist L_1 nicht trivial, da es `WHILE`-Programme gibt, die Funktionen ohne Nullstellen berechnen (z.B. $x \mapsto x + 1$) , als auch `WHILE`-Programme, die z.B die konstante 0 Funktion berechnen. Aus dem Satz von Rice folgt daher, dass $L_1 \notin \mathsf{REC}$ (2) . Desweiteren ist L_1 rekursiv aufzählbar (3). Das folgende Programm berechnet χ'_{L_1}:

Gegeben i, simulieren wir für alle Paare (m, t) das Programm i für t Schritte auf dem Input m. Wenn eine Simulation 0 ausgibt, geben wir 1 aus.

Da L_1 also nicht entscheidbar, aber rekursiv aufzählbar ist, kann das Komplement von L_1 nicht ebenfalls rekursiv aufzählbar sein (4).

(b) L_2 ist keine Indexmenge (1). Wir betrachten dazu die Funktion $h(x, y)$ welche $(x+1)^2$ ausgibt, wenn $x = y$ und 0 sonst. Da h offensichtlich `WHILE`-berechenbar ist , liefert uns das Rekursionstheorem dann die Existenz einer Gödelisierung i, so dass $\varphi_i(y) = h(i, y)$. Dann gilt $\varphi_i(i) = (i + 1)^2$, also $i \in L_2$. Sei nun $P = \text{göd}^{-1}(i)$. Wir fügen P ein simple statement hinzu, welches die von P berechnete Funktion nicht verändert, und erhalten P' . Nun gilt, dass $\varphi_{\text{göd}(P')} = \varphi_i$. Da aber $\text{göd}(P') \neq i$ gilt auch, dass $\varphi_{\text{göd}(P')}(\text{göd}(P')) = 0 \neq (\text{göd}(P') + 1)^2$. Daher ist $\text{göd}(P') \notin L_2$. Trotzdem ist L_2 nicht entscheidbar (2). Wir reduzieren dafür vom speziellen Halteproblem H_0 mit der Reduktionsfunktion

$$f(i) := \text{göd}(P_i)$$

wobei P_i das folgende Programm ist:

Gegeben m, simuliere i auf i und gib danach $(m + 1)^2$ aus.

Wenn $i \in H_0$ gilt dann, dass $\varphi_{f(i)}(f(i)) = \varphi_{\text{göd}(P_i)}(f(i)) = (f(i)+1)^2$, also $f(i) \in L_2$, da die Simulation von i auf i hält.

Wenn $i \notin H_0$ ist, dann hält i nicht auf i. Dann ist P_i überall undefiniert und folglich $f(i) = \text{göd}(P_i) \notin L_2$.

Da f `WHILE`-berechenbar ist und $H_0 \notin$ co-RE, ist also auch $L_2 \notin$ co-RE, also $L_2 \notin$ REC.

Desweiteren ist L_2 rekursiv aufzählbar (3). Das folgende Programm berechnet χ'_{L_2} :

Gegeben i, simulieren wir i auf i. Wenn die Simulation $(i + 1)^2$ ausgibt, geben wir 1 aus. Bei einer anderen Ausgabe divergieren wir.

Da L_2 also nicht entscheidbar, aber rekursiv aufzählbar ist, kann das Komplement von L_2 nicht ebenfalls rekursiv aufzählbar sein (4).

Aufgabe A9.4 (Best of Computability Theory 2) *(8 Bonuspunkte)*
Entscheiden Sie für jede der folgenden Mengen ob diese (1) eine Indexmenge, (2) entscheidbar, (3) rekursiv aufzählbar und/oder (4) co-rekursiv aufzählbar sind. Beweisen Sie Ihre Antworten.

(a) $L_3 = \{i \in \mathbb{N} \mid \text{im}\, \varphi_i = \emptyset\}$

(b) $L_4 = \{i \in \mathbb{N} \mid \varphi_i \text{ ist surjektiv}\}$

Lösung A9.4 (Best of Computability Theory 2)

(a) L_3 ist eine Indexmenge (1): Wenn $i \in L_3$ und $j \in \mathbb{N}$ mit $\varphi_j = \varphi_i$, dann ist das Bild von φ_j ebenfalls leer. Ferner ist L_3 nicht trivial, da es `WHILE`-Programme gibt, die Funktionen die ein nicht-leeres Bild haben berechnen (z.B. $f : x \mapsto x$ mit $\text{im}\, f = \mathbb{N}$) und zusätzlich gilt $\Omega \in L_3$, da $\text{im}\, \varphi_\Omega = \emptyset$. Aus dem Satz von Rice folgt daher, dass $L_3 \notin$ REC (2). Desweiteren ist $\overline{L_3}$ rekursiv aufzählbar (4). Das folgende Programm berechnet $\chi'_{\overline{L_3}}$:

Gegeben i, Simuliere für alle Paare (m, t) das Programm i für t Schritte auf den Input m. Wenn eine Simulation eine Ausgabe hat, geben wir 1 aus.

Da L_3 also nicht entscheidbar, aber co-rekursiv aufzählbar ist, kann L_3 nicht gleichzeitig rekursiv aufzählbar sein (3).

(b) L_4 ist eine Indexmenge (1): Wenn $i \in L_4$ und $j \in \mathbb{N}$ mit $\varphi_j = \varphi_i$, dann ist φ_j ebenfalls surjektiv[2]. Wir zeigen nun, dass $H_0 \leq L_4 (\Leftrightarrow \overline{H_0} \leq \overline{L_4})$ und $H_0 \leq \overline{L_4} (\Leftrightarrow \overline{H_0} \leq L_4)$ was impliziert, dass L_4 weder entscheidbar , noch rekursiv aufzählbar oder co-rekursiv aufzählbar ist (2),(3),(4), da $\overline{H_0} \notin \mathsf{RE}$.

$H_0 \leq L_4$

Sei

$$f(i) := \text{göd}(P_i)$$

wobei P_i das folgende Programm ist:

Gegeben m, simuliere i auf i und gib danach m aus.

f ist offensichtlich `WHILE`-berechenbar.

Wenn $i \in H_0$ gilt dann, dass $\varphi_{f(i)}(m) = \varphi_{\text{göd}(P_i)}(m) = m$, da die Simulation von i auf i hält. Also ist $\varphi_{f(i)}$ surjektiv und daher $f(i) \in L_4$.

Wenn $i \notin H_0$ hält i nicht auf i. Dann ist P_i überall undefiniert und folglich ist $\varphi_{\text{göd}(P_i)}$ nicht surjektiv, also $f(i) = \text{göd}(P_i) \notin L_4$.

$H_0 \leq \overline{L_4}$

Wir betrachten die Reduktionsfunktion

$$f(i) := \text{göd}(P_i)$$

wobei P_i das folgende Programm ist:

Gegeben m, simuliere i auf i für m Schritte. Wenn die Simulation nicht innerhalb der m Schritte terminiert hat, gib m aus. Ansonsten: Divergiere.

f ist offensichtlich `WHILE`-berechenbar.

Wenn $i \in H_0$ dann gibt es ein $c \in \mathbb{N}$, so dass i nach c Schritten auf i hält. Es folgt dann, dass $c \notin \text{im}\, \varphi_{\text{göd}(P_i)}$, also ist $\varphi_{f(i)} = \varphi_{\text{göd}(P_i)}$ nicht surjektiv und folglich $f(i) \in \overline{L_4}$.

Wenn $i \notin H_0$ hält i nicht auf i. Dann berechnet P_i die Identitätsfunktion und folglich ist $\varphi_{\text{göd}(P_i)}$ surjektiv, also $f(i) = \text{göd}(P_i) \notin \overline{L_4}$.

Aufgabe A9.5 (Rekursionstheorem) *(4 Bonuspunkte)*
Zeigen Sie, dass die folgenden Mengen nicht leer sind:

[2] Ferner ist L_4 nicht trivial, also könnten wir den Satz von Rice anwenden. Da aber, wie sich herausstellen wird, L_4 weder rekursiv noch co-rekursiv aufzählbar ist, reduzieren wir ohnehin vom Halteproblem, was Unentscheidbarkeit impliziert.

(a) $N_1 = \{i \in \mathbb{N} \mid \varphi_i(m) = i \cdot m\}$

(b) $N_2 = \{i \in \mathbb{N} \mid \varphi_i(42) = i^{1337}\}$

(c) $N_3 = \{\langle i, j\rangle \mid i, j \in \mathbb{N} \text{ und } i \neq j \text{ und } \varphi_i(j) + j = \varphi_j(i) + i\}$

Lösung A9.5 (Rekursionstheorem)
Wir benutzen für alle Aufgabenteile das Rekursionstheorem.

(a) Sei $f(g, m) = g \cdot m$. f ist offensichtlich `WHILE`-berechenbar , daher gibt es ein i mit $\varphi_i(m) = f(i, m) = i \cdot m$. Somit ist $i \in N_1$.

(b) Sei $f(g, m) = g^{1337}$. f ist offensichtlich `WHILE`-berechenbar , daher gibt es ein i mit $\varphi_i(42) = f(i, 42) = i^{1337}$. Somit ist $i \in N_2$.

(c) Sei $f_1(a, b) = a$ und $f_2(a, b) = 0$ wenn $b = 0$ und $f_2(a, b) = a$ sonst. Beide Funktionen sind offensichtlich `WHILE`-berechenbar , daher gibt es i und j (man beachte, dass sowohl i als auch j ungleich 0 sind, da das Programm mit der Gödelnummer 0 die Funktion $x \mapsto 2x$ berechnet), so dass

$$\varphi_i(j) + j = f_1(i, j) + j = i + j = j + i = f_2(j, i) + i = \varphi_j(i) + i$$

Ferner sind $i \neq j$, da $\varphi_i(0) = i \neq 0 = \varphi_j(0)$. Somit ist $\langle i, j\rangle \in N_3$.

Aufgabe A9.6 (Transduktoren als Reduktionen auf REG ($\star$)) *(4 Bonuspunkte)*
Wir hatten schon gezeigt, dass sich Regularität unter Transduktoren erhält, nun möchten wir zeigen, dass wir Transduktoren als Reduktionen auf REG betrachten können. Hierzu sagen wir für zwei Sprachen $L, L' \subseteq \Sigma^\star$ dass sich L auf L' mit einem Transduktor reduziert (In kurz: $L \leq_\mathrm{T} L'$), wenn ein deterministischer Transduktor[3] M existiert, mit totalem f_M und $x \in L \Leftrightarrow f_M(x) \in L'$.

Wir zeigen nun, dass dies eine sinvolle Definition von Reduktionen auf REG ist. Beweisen Sie hierzu die folgenden Eigenschaften:

(a) $\leq_\mathrm{T}$ ist reflexiv, d.h. für jede Sprache $L \subseteq \Sigma^\star$ gilt $L \leq_\mathrm{T} L$.

(b) $\leq_\mathrm{T}$ ist transitiv, d.h. für alle Sprachen $L, L', L'' \subseteq \Sigma^\star$ mit $L \leq_\mathrm{T} L'$ und $L' \leq_\mathrm{T} L''$ gilt $L \leq_\mathrm{T} L''$.

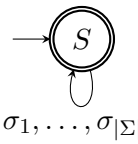
(c) Für alle Sprachen $L, L' \subseteq \Sigma^\star$ mit $L \leq_\mathrm{T} L'$ und $L' \in \mathsf{REG}$ gilt $L \in \mathsf{REG}$.

(d) Für alle Sprachen $L, L' \subseteq \Sigma^\star$ mit $L \leq_\mathrm{T} L'$ und $L \notin \mathsf{REG}$ gilt $L' \notin \mathsf{REG}$.

Lösung A9.6 (Transduktoren als Reduktionen auf REG ($\star$))

[3] Wir beschränken uns der Einfachheit halber hierbei auf das gleiche Ein-und Ausgabealphabet für den Transduktor, man kann aber einfach zeigen, dass dies keine tatsächliche Einschränkung ist.

(a) Reflexivität von $\leq_{\mathrm{T}}$ zeigen wir, indem wir einen Transduktor M angeben, für den gilt $f_M(x) = x$ für alle $x \in \Sigma^\star$. Daraus folgt dann direkt $L \leq_{\mathrm{T}} L$. Hierzu schreiben wir $\Sigma = \{\sigma_1, \ldots, \sigma_{|\Sigma|}\}$.



$\sigma_1 : \sigma_1, \ldots, \sigma_{|\Sigma|} : \sigma_{|\Sigma|}$

(b) Seien $M_1 = \{Q_1, \Sigma, \Sigma, \delta_1, f_1, q_{0,1}, Q_{\mathrm{acc},1}\}$ und $M_2 = \{Q_2, \Sigma, \Sigma, \delta_2, f_2, q_{0,2}, Q_{\mathrm{acc},2}\}$ totale Transduktoren mit $x \in L \Leftrightarrow f_{M_1}(x) \in L'$ und $x \in L' \Leftrightarrow f_{M_2}(x) \in L''$. Wir konstruieren nun explizit einen totalen Transduktor $M = \{Q, \Sigma, \Sigma, \delta, f, q_0, Q_{\mathrm{acc}}\}$ für den für alle $x \in \Sigma^\star$ gilt $f_M(x) = f_{M_2}(f_{M_1}(x))$. Daraus folgt direkt $f_M(L) = f_{M_2}(f_{M_1}(L)) = f_{M_2}(L') = L''$ und somit $L \leq_{\mathrm{T}} L''$. Intuitiv führen wir beide Transduktoren zeitgleich aus, dies zeichnet sich auch in der Konstruktion aus, die sehr ähnlich zu einem Produktautomaten ist:

$$
\begin{aligned}
Q &= Q_1 \times Q_2 \\
q_0 &= (q_{0,1}, q_{0,2}) \\
Q_{\mathrm{acc}} &= Q_{\mathrm{acc},1} \times Q_{\mathrm{acc},2} \\
\delta((q_1, q_2), \sigma) &= (\delta_1(q_1, \sigma), \delta_2^\star(q_2, f_1(q_1, \sigma))) && \text{für alle } q_1 \in Q_1, q_2 \in Q_2 \text{ und } \sigma \in \Sigma \\
f((q_1, q_2), \sigma) &= f_2^\star(q_2, f_1(q_1, \sigma)) && \text{für alle } q_1 \in Q_1, q_2 \in Q_2 \text{ und } \sigma \in \Sigma
\end{aligned}
$$

Um die Korrektheit zu zeigen, beweisen wir zuerst per Induktion über $|x|$, dass

$$f^\star((q_1, q_2), x) = f_2^\star(q_2, f_1^\star(q_1, x))$$

und

$$\delta^\star((q_1, q_2), x) = (\delta_1^\star(q_1, x), \delta_2^\star(q_2, f_1^\star(q_1, x)))$$

für alle $q_1 \in Q_1, q_2 \in Q_2, x \in \Sigma^\star$ gelten.

I.A. Sei $x = \varepsilon$ und $q_1 \in Q_1, q_2 \in Q_2$ beliebig. Hier gilt direkt

$$f^\star((q_1, q_2), \varepsilon) = \varepsilon = f_2^\star(q_2, \varepsilon) = f_2^\star(q_2, f_1^\star(q_1, \varepsilon))$$

und

$$
\begin{aligned}
\delta^\star((q_1, q_2), \varepsilon) &= (q_1, q_2) \\
&= (\delta_1^\star(q_1, \varepsilon), \delta_2^\star(q_2, \varepsilon)) \\
&= (\delta_1^\star(q_1, \varepsilon), \delta_2^\star(q_2, f_1^\star(q_1, \varepsilon)))\,.
\end{aligned}
$$

I.V. Gelte

$$f^\star((q_1, q_2), x') = f_2^\star(q_2, f_1^\star(q_1, x'))$$

und

$$\delta^\star((q_1, q_2), x') = (\delta_1^\star(q_1, x'), \delta_2^\star(q_2, f_1^\star(q_1, x')))$$

für alle $q_1 \in Q_1, q_2 \in Q_2$ und ein $x' \in \Sigma^\star$.

I.S. Sei $x = \sigma x'$ und $q_1 \in Q_1, q_2 \in Q_2$ beliebig. Es gilt

$$\begin{aligned} f^\star((q_1, q_2), \sigma x') &= f((q_1, q_2), \sigma) f^\star(\delta((q_1, q_2), \sigma), x') \\ &= f_2^\star(q_2, f_1(q_1, \sigma)) f^\star((\delta_1(q_1, \sigma), \delta_2^\star(q_2, f_1(q_1, \sigma))), x') \\ &\underset{\text{IV}}{=} f_2^\star(q_2, f_1(q_1, \sigma)) f_2^\star(\delta_2^\star(q_2, f_1(q_1, \sigma)), f_1^\star(\delta_1(q_1, \sigma), x')) \\ &= f_2^\star(q_2, f_1(q_1, \sigma) f_1^\star(\delta_1(q_1, \sigma), x')) \\ &= f_2^\star(q_2, f_1^\star(q_1, \sigma x')) \end{aligned}$$

und

$$\begin{aligned} \delta^\star((q_1, q_2), \sigma x') &= \delta^\star(\delta((q_1, q_2), \sigma), x') \\ &= \delta^\star((\delta_1(q_1, \sigma), \delta_2^\star(q_2, f_1(q_1, \sigma))), x') \\ &\underset{\text{IV}}{=} (\delta_1^\star(\delta_1(q_1, \sigma), x'), \delta_2^\star(\delta_2^\star(q_2, f_1(q_1, \sigma)), f_1^\star(\delta_1(q_1, \sigma), x') \\ &= (\delta_1^\star(q_1, \sigma x'), \delta_2^\star(q_2, f_1(q_1, \sigma) f_1^\star(\delta_1(q_1, \sigma), x') \\ &= (\delta_1^\star(q_1, \sigma x'), \delta_2^\star(q_2, f_1(q_1, \sigma x'))\,. \end{aligned}$$

Hierbei haben wir verwendet, dass die Aussagen $f_1^\star(q_1, yz) = f_1^\star(q_1, y) f_1^\star(\delta_1^\star(q_1, y), z)$ und $f_2^\star(q_1, yz) = f_2^\star(q_2, y) f_2^\star(\delta_2^\star(q_2, y), z)$ für alle $q_1 \in Q_1, q_2 \in Q_2$ und $y, z \in \Sigma^\star$ gelten. Dies lässt sich leicht mit einer weiteren Induktion beweisen.

Für beliebige $x \in \Sigma^\star$ akzeptiert M, genau dann, wenn $\delta^\star((q_{0,1}, q_{0,2}), x) \in Q_{\text{acc}} = Q_{\text{acc},1} \times Q_{\text{acc},2}$. Dies ist äquivalent zu $\delta_1^\star(q_{0,1}, x) \in Q_{\text{acc},1}$ und $\delta_2^\star(q_{0,2}, f_1^\star(q_{0,1}, x)) \in Q_{\text{acc},2}$ und letztlich äquivalent zur Definiertheit von $f_{M_1}(x)$ und $f_{M_2}(f_{M_1}(x))$. Insbesondere folgt aus der Totalität von f_{M_1} und f_{M_2} nun also die Totalität von f_M. Weiterhin gilt $f_M(x) = f^\star((q_{0,1}, q_{0,2}), x) = f_2^\star(q_{0,2}, f_1^\star(q_{0,1}, x)) = f_{M_2}(f_{M_1}(x))$, womit die Aussage gezeigt ist.

(c) Für $L \leq_{\text{T}} L'$ gibt es einen totalen Transduktor M mit $x \in L \Leftrightarrow f_M(x) \in L'$. Dies ist durch die Totalität von f_M nun aber äquivalent zu $L = \{x \in \Sigma^\star \mid f_M(x) \in L'\} = f_M^{-1}(L')$. Die Aussage folgt nun direkt aus Aufgabe A3.4.

(d) Dies ist genau die Kontraposition der vorherigen Teilaufgabe.

Aufgabe A9.7 (Andere Gödelisierungen ($\star$)) *(4 Bonuspunkte)*
Wir betrachten nun den Effekt den verschiedene Gödelisierungen haben können. Sei hierzu $\text{göd}' : \mathcal{W} \to \mathbb{N}$ eine beliebige bijektive berechenbare[4] Gödelisierung.

(a) Zeigen Sie, dass es unabhängig von der Wahl von $\text{göd}'$ immer ein $e \in \mathbb{N}$ gibt mit

$$\varphi_{\text{göd}'^{-1}(e)} = \varphi_{\text{göd}'^{-1}(e+1)} .$$

(b) Zeigen Sie, dass es jedoch von der Wahl von $\text{göd}'$ abhängt, ob es ein ein $e \in \mathbb{N}$ gibt mit

$$\varphi_{\text{göd}'^{-1}(e)} = \varphi_{\text{göd}'^{-1}(e+1)} = \varphi_{\text{göd}'^{-1}(e+2)} .$$

Zeigen Sie hierzu sowohl die Existenz einer Gödelisierung $\text{göd}'$ für die die Eigenschaft gilt, als auch eine für die diese nicht gilt.

(c) Zeigen Sie, dass unabhängig von der Wahl von $\text{göd}'$ die Menge

$$A = \{x \in \mathbb{N} \mid \varphi_{\text{göd}'^{-1}(x)} = \varphi_{\text{göd}'^{-1}(x+1)}\}$$

niemals eine Indexmenge über $\text{göd}'$ sein kann.

(d) Wir betrachten die Menge

$$B = \{x \in \mathbb{N} \mid \varphi_{\text{göd}'^{-1}(x)} = \varphi_{\text{göd}'^{-1}(x^2)}\} .$$

Zeigen Sie, dass B abhängig von der Wahl von $\text{göd}'$ nun sowohl eine Indexmenge über $\text{göd}'$ sein kann, als auch keine Indexmenge über $\text{göd}'$ sein kann.

Lösung A9.7 (Andere Gödelisierungen ($\star$)) Für alle Teilaufgaben überlegen wir uns, wie wir, gegeben eine unter göd berechenbaren[5], möglicherweise abzählbar unendlichen, Partitionierung $\mathcal{P}_1, \mathcal{P}_2, \ldots$ aller `WHILE`-Programme und eine berechenbare[6] Partitionierung $N_1, N_2, \ldots$ der natürlichen Zahlen mit $|\mathcal{P}_i| = |N_i|$ für alle i, eine berechenbare bijektive Gödelisierung $\text{göd}'$ konstruieren können mit $\text{göd}'(P) \in N_i$ für $P \in \mathcal{P}_1$:

Gegeben eine Gödelisierung $g = \text{göd}(P)$, berechne i mit $P \in \mathcal{P}_i$. Zähle die Anzahl k an Gödelisierungen $g' < g$ mit $\text{göd}^{-1}(g') \in \mathcal{P}_i$. Finde nun die k-te natürliche Zahl n, für die gilt $n \in N_i$ und gib diese als $\text{göd}'(P)$ zurück.

Das so konstruierte $\text{göd}'$ ist nun direkt eine berechenbare bijektive Gödelisierung, da sich das Verfahren analog umkehren lässt um, gegeben $\text{göd}'(P)$ ebenfalls $\text{göd}(P)$ zu finden.

(a) Die Funktion $f(g) = g + 1$ ist offensichtlich `WHILE`-berechenbar. Nun gibt es nach dem Fixpunktsatz ein $e \in \mathbb{N}$ mit $\varphi_e(x) = \varphi_{f(g)}$. Der Fixpunktsatz basiert hierbei nicht auf unserer genauen Gödelisierung, sondern lediglich auf der Berechenbarkeit dieser.

[4] Berechenbar heißt formal hierbei, dass es eine `WHILE`-berechenbare bijektive Funktion $f : \mathbb{N} \to \mathbb{N}$ gibt mit $\text{göd}' = f \circ \text{göd}$. Diese Eigenschaft muss nicht explizit bewiesen werden.
[5] das heißt hier: Gegeben $\text{göd}(P)$ ist es berechenbar $i \in \mathbb{N}$ mit $P \in \mathcal{P}_i$ zu finden
[6] das heißt hier: Gegeben $n \in \mathbb{N}$ is es berechenbar $i \in \mathbb{N}$ mit $n \in N_i$ zu finden.

(b) Wir konstruieren zuerst ein göd′ für das die Aussage nicht gilt. Wir definieren uns göd′ aus den folgenden Partitionierung aller `WHILE`-Programme und natürlichen Zahlen:

$$\begin{aligned}
\mathcal{P}_0 &:= \{\text{Programme der Form } [x_0 := 0; x_1 := c] \text{ für } c \in \mathbb{N}\} \\
\mathcal{P}_1 &:= \{\text{Programme der Form } [x_0 := 1; x_1 := c] \text{ für } c \in \mathbb{N}\} \\
\mathcal{P}_2 &:= \mathcal{W} \setminus (\mathcal{P}_1 \cup \mathcal{P}_2) \\
N_0 &:= \{3n \mid n \in \mathbb{N}\} \\
N_1 &:= \{3n+1 \mid n \in \mathbb{N}\} \\
N_2 &:= \{3n+2 \mid n \in \mathbb{N}\}
\end{aligned}$$

Nun berechnen alle Programme in $\mathcal{P}_0$ und $\mathcal{P}_1$ jeweils unterschiedliche konstante Funktionen. Für drei konsekutive Gödelnummern unter göd′ ist nun aber mindestens eine Gödelnummer in N_0 und eine in N_1, diese entsprechen dann einem Programm aus $\mathcal{P}_0$ und einem aus $\mathcal{P}_1$, sie können also insbesondere nicht alle die gleiche Funktion berechnen.

Nun zeigen wir weiterhin, dass es aber andere Gödelisierungen göd′ gibt, für die die Aussage gilt. Wir konstruieren diese wieder direkt anhand von Partitionen:

$$\begin{aligned}
\mathcal{P}_0 &:= \{x_1 := 1, x_1 := 2, x_1 := 3\} \\
\mathcal{P}_1 &:= \mathcal{W} \setminus \mathcal{P}_0 \\
N_0 &:= \{0, 1, 2\} \\
N_1 &:= \mathbb{N} \setminus N_0
\end{aligned}$$

Nun gilt für $e = 0$ direkt $\varphi_e(x) = \varphi_{e+1}(x) = \varphi_{e+2}(x) = x$ für alle $x \in \mathbb{N}$.

(c) Angenommen A wäre eine Indexmenge. Nach Teilaufgabe (a) wissen wir, dass A nicht leer ist. Sei also $g \in A$. Dann gilt $\varphi_g = \varphi_{g+1}$. Somit ist ebenfalls $g + 1 \in A$. Nun gilt aber induktiv weiter, dass $g + i \in A$ für alle $i \in \mathbb{N}$. Nun würde dies aber implizieren, dass es nur endlich viele Programme gibt, die eine andere Funktion als φ_g berechnen, dies ist aber ein Widerspruch, denn es gibt abzählbar unendlich viele `WHILE`-berechenbare Funktionen (zB. eine für jede konstante Funktion).

(d) Wir konstruieren zuerst ein göd′ für das die Aussage gilt. Wir definieren uns göd′ aus den folgenden Partitionierung aller `WHILE`-Programme und natürlichen Zahlen

(hierbei ist $i >= 1$):

$$\begin{aligned}
\mathcal{P}_0 &:= \{\text{Programme der Form } x_1 := c \text{ für } c \in \mathbb{N} \\
\mathcal{P}_i &:= \{\text{Programme der Form } [\text{göd}^{-1}(i-1); x_1 := c] \text{ für } c \in \mathbb{N}\} \\
&\quad \cup \{i-1 \mid \text{Falls göd}^{-1}(i-1) \text{ weder die Form } x_1 := c, \\
&\qquad\qquad \text{noch die Form } [P; x_1 := c] \text{ für irgendein } P \in \mathcal{W} \text{ und } c \in \mathbb{N} \text{ hat}\} \\
N_0 &:= \{n \mid n \text{ ist keine Primzahlpotenz}\} \\
N_i &:= \{p_i^k \mid k \in \mathbb{N} \text{ und } p_i \text{ ist die } i\text{-te Primzahl}\}
\end{aligned}$$

Wir bemerken nun, dass für alle $i \in \mathbb{N}$ alle Programme innerhalb von $\mathcal{P}_i$ die gleiche Funktion berechnen. Weiterhin ist für $j \in N_i$ für $i \in \mathbb{N}$ ebenfalls $j^2 \in N_i$, es gilt also für jede Gödelisierung $g \in \mathbb{N}$ direkt schon $\varphi_g = \varphi_{g^2}$ und es gilt $B = \mathbb{N}$, die Menge B ist also eine triviale Indexmenge.

Nun zeigen wir weiterhin, dass es aber andere Gödelisierungen göd$'$ gibt, für die die Aussage nicht gilt. Wir konstruieren diese wieder direkt anhand von Partitionen:

$$\begin{aligned}
\mathcal{P}_0 &:= \{x_1 := 0, x_1 := 1, x_0 := 0\} \\
\mathcal{P}_1 &:= \mathcal{W} \setminus \mathcal{P}_0 \\
N_0 &:= \{2, 4, 16\} \\
N_1 &:= \mathbb{N} \setminus N_0
\end{aligned}$$

Nun gilt für $i = 2$ direkt $\varphi_i(x) = \varphi_{i^2}(x) = x$ für alle $x \in \mathbb{N}$, also $i \in B$. Weiterhin gilt für $i = 4$ direkt $\varphi_i(x) = x \neq 0 = \varphi_{i^2}(x)$ für alle $x \in \mathbb{N} \setminus \{0\}$, also $i \notin B$. Es gilt aber $\varphi_2 = \varphi_4$, somit kann B keine Indexmenge sein.

Aufgabe A9.8 (Programmanalyse ($\star\star$)) *(4 Bonuspunkte)*
In dieser Aufgabe zählen wir als Anzahl Schritte eines Programmes nur die Anzahl an ausgeführten simple statements.

Wir definieren:

$$L = \{\langle g, t\rangle \in \mathbb{N} \mid \text{es gibt eine Eingabe } m \in \mathbb{N}, \text{ so dass } g \text{ auf } m \text{ in} \leq t \text{ Schritten terminiert}\}.$$

Zeigen oder widerlegen Sie: $L \in \mathsf{REC}$.

Lösung A9.8 (Programmanalyse ($\star\star$))
Wir zeigen, dass $L \in \mathsf{REC}$.

Wir betrachten zuerst was passiert, wenn wir eine "sehr große" Eingabe m in das Programm g eingeben und g für t Schritte ausführen. Was in dieser Aufgabe "sehr groß" bedeutet werden wir später noch betrachten.

Dann durchläuft das Programm g bei Eingabe m eine gewisse Folge von $\leq t$ simple statements und terminiert danach entweder, oder hält innerhalb der t Schritte noch nicht. Nun behaupten wir $m+1$ durchläuft exakt die selben simple statements und wird somit g zum Halten bringen genau dann, wenn m g zum Halten bringt.

Da wir nur eine einzelne Ausführung betrachten können wir oBdA. annehmen, dass jede Variable nur ein einziges Mal gesetzt wird. Da m sehr groß ist, ist jede Variable entweder eine Konstante c unabhängig von m, oder ein positives ganzzahliges Vielfaches von m plus oder minus einer Konstanten unabhängig von m zugewiesen. Diese beiden Fälle lassen sich zusammenfassen zu: jeder Variable ist ein nicht-negatives Vielfaches von m plus einer ganzzahligen Konstanten unabhängig von m zugewiesen.

Dies lässt sich per Induktion über die Abfolge von simple statements zeigen:

I.A. Vorbelegung der Variablen: Die Variable x_0 ist am Anfang auf den Wert m gesetzt, dies ist offensichtlich ein nicht-negatives ganzzahliges Vielfaches von m, alle weiteren Variablen sind mit 0 initialisiert, also $\sigma_i = 0 \cdot m + 0$ für alle $i > 0$.

I.S. "$P; x_i := c$": Die Variable x_i ist auf eine Konstante c unabhängig von m gesetzt, also $\sigma_i = 0 \cdot m + c$.

I.S. "$P; x_i := x_j + x_k$": Wenn $\sigma_j = a_1 \cdot m + c_1$ und $\sigma_k = a_2 \cdot m + c_2$ gilt, dann gilt $\sigma_i = (a_1 + a_2) \cdot m + (c_1 + c_2)$.

I.S. "$P; x_i := x_j - x_k$": Wenn $\sigma_j = a_1 \cdot m + c_1$ und $\sigma_k = a_2 \cdot m + c_2$ gilt, gibt es zwei Fälle:

- $a_1 \geq a_2$*:* Hier gilt direkt $\sigma_i = (a_1 - a_2) \cdot m + (c_1 - c_2)$, wobei $a_1 - a_2 \geq 0$.
- $a_1 < a_2$*:* In diesem Fall gilt $\sigma_i = 0 \cdot m + 0$, da wir m als groß genug annehmen und somit σ_k viel größer als σ_j ist.

Sei c nun der größte Betrag einer Konstanten die in diesem induktiven Beweis vorkommt. Diese ist für alle möglichen Ausführungspfade durch $2^t \cdot c'$ beschränkt, wobei c' die größte in g vorkommende Konstante ist. Dies lässt sich leicht per Induktion zeigen. Dann gilt für $m > c$, dass m groß genug ist, dass die Induktion funktioniert. Weiterhin sind für m und $m + 1$ die Werte aller Variablen entweder in beiden Fällen $= 0$ oder in beiden Fällen $\neq 0$. Daraus folgt dann, dass auf beiden Eingaben das Programm sich identisch verhält und entweder in beiden Fällen innerhalb von t Schritten terminiert oder in keinem. Das Programm verhält sich also für alle Eingaben $> c$ gleich, es reicht daher nur die Eingaben $\leq c + 1$ zu prüfen.

Ein Entscheider für L ist nun also der folgende:

Bei Eingabe $\langle g, t \rangle$ bestimme die größte in g vorkommende Konstante c' und simuliere g für t Schritte für alle Eingaben $\leq 2^t \cdot c' + 1$. Falls eine dieser Simulationen terminiert, gib 1 aus, ansonsten 0.

Aufgabe A9.9 (Satz von Rice für RE ($\star\star$)) *(4 Bonuspunkte)*

(a) Sei I eine Indexmenge. Zeigen Sie: Ist I rekursiv aufzählbar, dann gibt es für jedes $g \in I$ ein $e \in I$ mit $\mathrm{dom}(\varphi_e) \subseteq \mathrm{dom}(\varphi_g)$ und $\mathrm{dom}(\varphi_e)$ ist endlich.

(b) Folgern Sie aus (a), dass $T = \{g \in \mathbb{N} \mid \varphi_g \text{ ist total}\}$ nicht rekursiv aufzählbar ist.

(c) Zeigen Sie, dass auch $\overline{T}$ nicht rekursiv aufzählbar ist.

Lösung A9.9 (Satz von Rice für RE ($\star\star$))

(a) Sei $k \in \mathbb{N}$ die Gödelnummer eines Programmes mit $\varphi_k(x) = 1$ für $x \in I$ und $\varphi_k(x)$ undefiniert sonst. Sei eine Funktion f gegeben durch

$$f(j,n) = \begin{cases} \varphi_g(n) & \text{falls } \varphi_k(j) \text{ nicht in } n \text{ Schritten hält und} \\ \text{undefiniert} & \text{sonst.} \end{cases}$$

Da f offensichtlich `WHILE`-berechenbar ist gibt es nach dem Rekursionstheorem ein $e \in \mathbb{N}$ mit $\varphi_e(n) = f(e,n)$.

Angenommen, es wäre $e \notin I$. Dann ist $\varphi_k(e)$ undefiniert und damit $f(e,n) = \varphi_g(n)$ für alle n. Daraus folgt $\varphi_e = \varphi_g$. Weil I eine Indexmenge ist, muss $e \in I$ gelten – Widerspruch. Also ist $e \in I$.

Aus $e \in I$ folgt, dass $\varphi_k(e)$ definiert ist. Damit ist $f(e,n)$ nur für endliche viele n definiert. Also ist $\mathrm{dom}(\varphi_e)$ endlich. Ist $\varphi_e(n)$ definiert, dann ist $\varphi_e(n) = \varphi_g(n)$, also ist $\mathrm{dom}(\varphi_e)$ eine endliche Teilmenge von $\mathrm{dom}(\varphi_g)$.

(b) Wir hatten schon gezeigt, dass T eine Indexmenge ist. T enthält keine Indizes von Funktionen mit endlichem Definitionsbereich, aber welche mit unendlichem Definitionsbereich. Damit ist T nicht rekursiv aufzählbar.

(c) Es gilt $\Omega \notin T$. Somit gilt nach Aufgabe P8.1, dass $H_0 \leq T$. Da $H_0 \notin \mathsf{co\text{-}RE}$, gilt also auch $T \notin \mathsf{co\text{-}RE}$.

Aufgabe A9.10 (Regularitätserhaltend ($\star\star\star$)) *(4 Bonuspunkte)*

(a) Sei $L \in \mathsf{REG}$. Zeigen Sie, dass die folgende Sprache ebenfalls regulär ist:

$$L_H = \{x \mid \exists y \in \Sigma^\star : xy \in L \text{ und } |x| = |y|\}$$

(b) Allgemein: eine Funktion $f \in \Sigma^\star : \mathbb{N} \to \mathbb{N}$ ist regularitätserhaltend, wenn für alle $L \in \mathsf{REG}$, die Sprache

$$L_f = \{x \mid \exists y : xy \in L \text{ und } f(|x|) = |y|\}$$

ebenfalls regulär ist. Zeigen Sie, dass f genau dann regularitätserhaltend ist, wenn für alle letztendlich periodischen[7] $U \subseteq \mathbb{N}$ gilt, dass $f^{-1}(U) = \{u \mid f(u) \in U\}$ ebenfalls letztendlich periodisch ist.

[7] letztendlich periodisch hatten wir in Aufgabe 2.3 auf dem Präsenzblatt 2 definiert.

Lösung A9.10 (Regularitätserhaltend ($\star\star\star$))

(a) Sei $M = (Q, \Sigma, \delta, q_0, Q_{\text{acc}})$ ein deterministischer endlicher Automat mit $L(M) = L$. Wir konstruieren einen neuen Automaten M', der M gleichzeitig vorwärts, beginnend mit dem Startzustand, und rückwärts, beginnend mit allen Endzuständen, simuliert. Er akzeptiert, falls der vorwärts erreichte Zustand gleich einem der rückwärts erreichten Zustände ist.

Sei $M' = (Q', \Sigma, \delta', (q_0, Q_{\text{acc}}), Q'_{\text{acc}})$, wobei die Menge der Zustände

$$Q' = Q \times \mathcal{P}(Q)$$

und die Menge der akzeptierenden Zustände

$$Q'_{\text{acc}} = \{(q, W) \mid q \in W \subseteq Q\} \subseteq Q'$$

ist. Die Zustandsübergangsfunktion $\delta : Q' \times \Sigma \to Q'$ ist durch

$$\delta'\big((q, W), x\big) = \big(\delta(q, x), W'\big)$$

mit

$$W' = \{q' \mid \exists y \in \Sigma : \ \delta(q', y) \in W\}$$

gegeben. Es bleibt zu beweisen, dass $L(M') = L_H$ ist.

$L_H \subseteq L(M')$: Sei $x \in L_H$ beliebig. Es gibt also $y \in \Sigma^\star$ und $\ell \in \mathbb{N}$ mit $|y| = |x| = \ell$ und $xy \in L$. Aus $xy \in L$ folgt, dass es Zustände $q_0, q_1, \ldots, q_{2\ell}$ gibt mit $\delta\big(q_i, (xy)_{i+1}\big) = q_{i+1}$ und $q_{2\ell} \in Q_{\text{acc}}$.

Wir zeigen nun, dass es in M' eine Folge $(q_0, Q_0), (q_1, Q_1), \ldots, (q_\ell, Q_\ell)$ von Zuständen mit $\delta'\big((q_{i-1}, Q_{i-1}), x_i\big) = (q_i, Q_i)$ gibt, so dass M' den String x akzeptiert, also $q_\ell \in Q_\ell$ ist. Hierbei sind $q_0, \ldots, q_\ell$ die Zustände wie oben, die M auf x durchläuft. Wir setzen $Q_0 = Q_{\text{acc}}$. Dann ist (q_0, Q_0) der Startzustand von M' und $q_{2\ell} \in Q_0$. Nun beweisen wir, dass $q_{2\ell-i} \in Q_i$ ist. Dann ist $x \in L(M')$, weil $q_\ell \in Q_\ell$ ist, also $(q_\ell, Q_\ell) \in Q'_{\text{acc}}$.

Sei $i \in \{0, \ldots, \ell - 1\}$ beliebig und (q_i, Q_i) gegeben, und sei $q_{2\ell-i} \in Q_i$ nach Voraussetzung. Dann ist $\delta(q_{2\ell-i-1}, y_{\ell-i}) = q_{2\ell-i}$ (siehe Berechnung von M auf xy), und somit $q_{2\ell-i-1} \in Q_{i+1}$.

$L_H \supseteq L(M')$: Sei $x \in L(M')$, $|x| = \ell$. Dann gibt es eine Folge $(q_0, Q_0), \ldots, (q_\ell, Q_\ell)$ von Zuständen mit $q_\ell \in Q_\ell$. Nach Lesen von x wäre M in Zustand q_ℓ. Wegen $q_\ell \in Q_\ell$ gibt es ein $q_{\ell+1} \in Q_{\ell-1}$ und $y_1 \in \Sigma$ mit $\delta(q_\ell, y_1) = q_{\ell+1}$. Analog gibt es für jedes $i \in \{1, \ldots, \ell\}$ ein $q_{\ell+i} \in Q_{\ell-i}$ und $y_i \in \Sigma$ mit $\delta(q_{\ell+i-1}, y_i) = q_{\ell+i}$. Auf diese Weise konstruieren wir einen String y, der ebenfalls Länge ℓ hat.

Nach Lesen von xy befindet sich M in einem Zustand $q_{2\ell} \in Q_0 = Q_{\text{acc}}$. Damit ist $xy \in L$ und $x \in L_H$.

(b) Wir benötigen für diese Aufgabe folgende beide Lemmata:

Lemma 1: Wenn $L \subseteq \Sigma^\star$ regulär ist, dann ist $\{|x| \mid x \in L\}$ letztendlich periodisch.

Lemma 2: Wenn $A \subseteq \mathbb{N}$ letztendlich periodisch ist, dann ist $\{x \in \Sigma^\star \mid |x| \in A\}$ regulär.

Beide Aussagen folgen direkt aus Aufgabe 2.3 des Präsenzblattes 2 unter Anwendung des Homomorphismus $h(a) = 1$ für alle $a \in \Sigma$.

Sei nun also f regularitätserhaltend und U eine letztendlich periodische Menge. Dann ist nach Lemma 2 die Sprache $L = \{1^n \mid n \in U\}$ regulär. Ebenfalls regulär ist dann aber auch

$$L' = \{x \mid \exists y : xy \in L(0^\star 1)L \text{ und } f(|x|) = |y|\} \cap L(0^\star 1)\,,$$

da f regularitätserhaltend erhaltend ist und $L(0^\star 1)L$ regulär ist.

Wir betrachten nun:

$$\begin{aligned}
A &= \{|x| \mid x \in L'\} \\
&= \{|x| \mid x \in L(0^\star 1) \text{ und } \exists y \in \Sigma^\star : xy \in L(0^\star 1)L \text{ und } f(|x|) = |y|\} \\
&= \{|x| \mid x \in L(0^\star 1) \text{ und } \exists y \in L : f(|x|) = |y|\} \\
&= \{|x| \mid n > 0 \text{ und } \exists n' \in U : f(n) = n'\} \\
&= \{|x| \mid n > 0 \text{ und } f(n) \in U\} \\
&= f^{-1}(U) \setminus \{0\}
\end{aligned}$$

Da L' regulär war, ist nach Lemma 1 sowohl A, als auch das an nur endlich vielen Stellen von A unterschiedliche $f^{-1}(U)$, letztendlich periodisch.

Gelte nun für alle letztendlich periodischen U, dass $f^{-1}(U)$ letztendlich periodisch ist und sei $L \subseteq \Sigma^\star$ eine beliebige reguläre Sprache. Wir verwenden das Myhill-Nerode Theorem und partitionieren $\Sigma^\star$ in die endlich vielen Myhill-Nerode-Äquivalenzklassen $L_1, L_2, \cdots, L_r$ bezüglich L.

Da die L_i eine Partition bilden gilt nun also

$$L_f = (L_f \cap L_1) \cup (L_f \cap L_2) \cup \cdots \cup (L_f \cap L_r)\,.$$

Wir zeigen die Regularität aller Mengen $L_f \cap L_i$ einzeln, woraus dann direkt die Regularität von L_f als endliche Vereinigung regulärer Sprachen folgt. Hierzu benötigen wir, dass alle $x \in L_i$ Myhill-Nerode äquivalent sind, es also eine Menge $R_i = \{y \in \Sigma^\star \mid xy \in L\}$ gibt, die nicht von der genauen Wahl von x abhängt, sondern nur von i. Sowohl L_i, als auch R_i sind regulär. Um dies zu sehen nehmen wir den minimalen DEA M, der L erkennt, Wenn wir genau den Zustand der zur Menge L_i gehört akzeptierend machen erkennen wir L_i, wenn wir ihn hingegen zum Startzustand machen erkennen wir R_i.

$$\begin{aligned}
L_f \cap L_i &= \{x \in \Sigma^\star \mid \exists y \in \Sigma^\star : xy \in L \text{ und } f(|x|) = |y|\} \cap L_i \\
&= \{x \in L_i \mid \exists y \in \Sigma^\star : xy \in L \text{ und } f(|x|) = |y|\}
\end{aligned}$$

da $x \in L_i$ läst sich die Bedingung $xy \in L$ durch $y \in R_i$ austauschen

$$\begin{aligned}
&= \{x \in L_i \mid \exists y \in R_i : f(|x|) = |y|\} \\
&= \{x \in \Sigma^\star \mid \exists y \in R_i : f(|x|) = |y|\} \cap L_i \\
&= \{x \in \Sigma^\star \mid |x| \in f^{-1}(\{|y| \mid y \in R_i\})\} \cap L_i
\end{aligned}$$

Nun ist aber nach Lemma 1 $\{|y| \mid y \in R_i\}$ letztendlich periodisch und somit ist $f^{-1}(\{|y| \mid y \in R_i\})$ ebenfalls letztendlich periodisch. Nach Lemma 2 ist nun also $\{x \in \Sigma^\star \mid |x| \in f^{-1}(\{|y| \mid y \in R_i\})\}$ regulär, woraus die Regularität von $L_f \cap L_i$ und somit L_f folgt.

Aufgabe A9.11 (Substrings ($\star\star\star$)) *(4 Bonuspunkte)*
Für ein beliebiges endliches Alphabet Σ und beliebige $x, y \in \Sigma^\star$ gilt $x \preceq y$ (x ist Substring von y), falls x aus y durch Löschen von null oder mehr Zeichen hervorgeht. (*Beispiel:* $\varepsilon \preceq 00 \preceq 0110 \preceq 0110 \preceq 001001001$.)

Eine Sprache $L \subseteq \Sigma^\star$ heißt *substring-abgeschlossen*, falls aus $y \in L$ und $x \preceq y$ auch $x \in L$ folgt.

(a) Zeigen Sie: Ist $A \subseteq \Sigma^\star$ unendlich, dann gibt es $x, y \in A$ mit $x \neq y$ und $x \preceq y$.

(b) Zeigen Sie: Jede substring-abgeschlossene Sprache $L \subseteq \Sigma^\star$ ist regulär.

Lösung A9.11 (Substrings ($\star\star\star$))

(a) Wir beweisen eine leicht stärkere Aussage: Für jede unendliche Liste A an Wörtern aus $\Sigma^\star$, existieren $i < j$ mit $A_i \preceq A_j$.

Wir nennen eine unendliche Liste *frei*, wenn diese Eigenschaft verletzt ist, es also keine $i < j$ mit $A_i \preceq A_j$ gibt. Angenommen es existieren freie Listen, dann können wir eine freie Liste A wie folgt konstruieren: Wähle als A_1 ein Wort minimaler Länge, so dass eine unendliche freie Liste existiert, die mit A_1 anfängt. Danach wähle ein A_2 minimaler Länge, so dass eine unendliche freie Liste existiert, die mit A_1, A_2 anfängt, usw.

Da A unendlich lang ist, muss es nun ein $\sigma \in \Sigma$ geben, so dass unendlich viele Worte in A mit σ anfangen. Sei das erste solche Wort A_k. Wir konstruieren nun eine neue Liste B wie folgt: $A_1, A_2, \ldots, A_{k-1}$. Danach enthält B nur solche Wörter aus A, die mit σ anfangen, jedoch ohne das erste Zeichen.

Nun ist B aber ebenfalls eine unendliche freie Liste. Es gelten nun die folgenden einfachen Äquivalenzen:

$$B_i \preceq B_j \Leftrightarrow B_j \preceq \sigma B_j \qquad \text{falls } (B_i)_1 \neq \sigma \tag{1}$$

$$B_i \preceq B_j \Leftrightarrow \sigma B_j \preceq \sigma B_j \tag{2}$$

Nehmen wir an es existieren nun $i < j$ mit $B_i \preceq B_j$.

$i < k$: Nun muss gelten $j \geq k$, da sonst $A_i \preceq A_j$ gelten würde da $A_i = B_i$ und $A_j = B_j$. B_i fängt nach Konstruktion nicht mit σ an, nach (1) gilt also $B_i \preceq \Sigma B_j$. Nun gilt aber $B_i = A_i$ und $\sigma B_j = A_{j'}$ für ein $j' > i$. Dies ist ein Widerspruch dazu, dass A frei ist.

$i \geq k$: Nach Konstruktion von B gibt es $i' < j'$ mit $A_{i'} = \sigma B_i$ und $A_{j'} = \sigma B_j$. Nach (2) gilt nun aber $\sigma B_i \preceq \sigma B_j$, erneut ein Widerspruch dazu, dass A frei ist.

Somit kann A nicht existieren, es existiert also insbesondere keine unendliche freie Liste.

(b) Ist L substring-abgeschlossen, dann folgt aus $x \in \overline{L}$ und $x \preceq y$ auch $y \in \overline{L}$.

Sei $X = \{x \in \overline{L} \mid \forall y \in \overline{L} : y \preceq x \Rightarrow x = y\}$ die Menge der minimalen Elemente aus $\overline{L}$. Wegen Teil (a) ist X endlich. Für $x = x_1 \dots x_n$ sei

$$L_x = L\big(\Sigma^\star x_1 \Sigma^\star x_2 \Sigma^\star \dots \Sigma^\star x_n \Sigma^\star\big) = \{y \mid x \preceq y\}.$$

Offensichtlich ist L_x regulär. Wir behaupten, dass $\overline{L} = \bigcup_{x \in X} L_x$ ist:

$\overline{L} \subseteq \bigcup_{x \in X} L_x$: Ist $y \in \overline{L}$, dann ist y entweder minimal, also $y \in X$, woraus $y \in \bigcup_{x \in X} L_x$ folgt. Oder es gibt ein minimales $z \in \overline{L}$, woraus $z \in X$ und $y \in L_z$ folgt.

$\overline{L} \supseteq \bigcup_{x \in X} L_x$: Wir zeigen $L_x \subseteq \overline{L}$. für ein beliebiges $x \in X$. Wegen $X \subseteq \overline{L}$ ist $x \in \overline{L}$. Sei $y \in L_x$ beliebig, dann ist $x \preceq y$, also auch $y \in \overline{L}$.

Damit ist $\overline{L}$ die endliche Vereinigung regulärer Sprachen, also selbst regulär. Folglich ist auch L regulär.