Technische Universität München
Fakultät für Informatik
Lehrstuhl Informatik XIV
Prof. Dr. Ernst W. Mayr
Dr. Werner Meixner

Sommersemester 2015
Lösungsblatt 13
17. Juli 2015

Theoretische Informatik

Hausaufgabe 1

1. Sei $f : \mathbb{N}_0 \to \mathbb{N}_0$ definiert durch die Startwerte $f(0) = 1$, $f(1) = 2$, $f(2) = 3$ zusammen mit der Rekursion

$$f(n) = f(n-1) \cdot f(n-3) \quad \text{für alle} \quad n \in \mathbb{N} \setminus \{1, 2\}.$$

Zeigen Sie die primitive Rekursivität der Funktion f, indem Sie die Erzeugungsregeln für primitiv-rekursive Funktionen zusammen mit einer Paarfunktion $p : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$ und deren Umkehrfunktionen c_1 und c_2 anwenden. Kodieren Sie dabei $h(n) = p(p(f(n), f(n+1)), f(n+2))$ für alle $n \in \mathbb{N}_0$.

Hinweis : Sie dürfen zusätzlich zu den Basisfunktionen der primitiven Rekursion die folgenden Funktionen als primitiv-rekursiv annehmen: $plus(m, n)$ $(+)$, $times(m, n)$ $(\cdot)$, $p(m, n)$ und $c_1(n)$, $c_2(n)$. Sie dürfen die erweiterte Komposition und das erweiterte rekursive Definitionsschema benützen. LOOP- und WHILE-Programme sind **nicht** erlaubt.

2. Wir betrachten die Menge

$$F = \{w \in \{0, 1\}^* ;\ M_w \text{ berechnet } f\}.$$

Sei $H_0 = \{w \in \{0, 1\}^* ; M_w \text{ hält auf leerem Band}\}$ das Halteproblem auf leerem Band. Zeigen Sie durch informelle Spezifikation einer Reduktionsabbildung g (wie in entsprechenden Beweisen der Vorlesung), dass H_0 reduzierbar ist auf F, i. Z. $H_0 \leq F$.

Lösung

1. Sei $h(n) = p(p(f(n), f(n+1)), f(n+2))$.

Dann gilt $h(0) = p(p(1, 2), 3)$.

Wegen $f(n+3) = f(n+2) \cdot f(n)$, $f(n+2) = c_2(h(n))$, $f(n+1) = c_2(c_1(h(n)))$, $f(n) = c_1(c_1(h(n)))$ gilt das Schema der primitiven Rekursion

$$\begin{aligned}
h(0) &= p(p(1, 2), 3), \\
h(n+1) &= p(p(f(n+1), f(n+2)), f(n+2) \cdot f(n)) \\
&= p(p(c_2(c_1(h(n))), c_2(h(n))), c_2(h(n)) \cdot c_1(c_1(h(n)))))
\end{aligned}$$

Da h primitiv rekursiv ist und $f(n) = c_1(c_1(h(n)))$ gilt, folgt die PR von f.

2. Sei w_f der Code einer Turingmaschine M_{w_f}, die f berechnet.

   Für alle $w \in \{0,1\}^*$ sei $g(w)$ der Code einer Turingmaschine M, die wie folgt definiert ist:

   M simuliert eine 2-Band-Turingmaschine, die die Eingabe von M auf Band 1 schreibt und anschließend auf Band 2 die Turingmaschine M_w auf leerem Band ausführt.

   Falls M_w hält, dann wird auf Band 1 die Eingabe von M mit M_{w_f} ausgeführt und das Ergebnis auf das Band von M geschrieben.

   Offenbar ist g eine totale und berechenbare Funktion, so dass $g(H_0) \subseteq F$ und $g(\overline{H_0}) \subseteq \overline{F}$ gelten.

Hausaufgabe 2

1. Sei $f : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$ für alle $m, n \in \mathbb{N}_0$ gegeben durch $f(m,n) = m^2 \dot{-} n$.

   Zeigen Sie, dass μf primitiv-rekursiv ist.

2. Sei $g : \mathbb{N}_0 \to \mathbb{N}_0$ total und μ-rekursiv, und sei $f : \mathbb{N}_0 \to \mathbb{N}_0$ definiert durch die Startwerte $f(0) = 1$ und $f(1) = 2$ zusammen mit der Rekursion

   $$f(n) = g(n) \cdot f(n-1) \cdot f(n-2) \quad \text{für alle} \quad n \in \mathbb{N}_0 \text{ mit } n \geq 2\,.$$

   Zeigen Sie durch Anwendung der Erzeugungsregeln für μ-rekursive Funktionen mit Hilfe der Paarfunktion $p : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$ und deren Umkehrfunktionen c_1 und c_2, dass f μ-rekursiv ist.

   Hinweis:
   Sie dürfen zusätzlich zu den Basisfunktionen der primitiven Rekursion die folgenden Funktionen als primitiv-rekursiv annehmen: $plus(m,n)$ $(+)$, $times(m,n)$ $(\cdot)$, $pred(n)$, $p(m,n)$, $c_1(n)$, $c_2(n)$, $ifthen(n,a,b)$ und die konstante k-stellige Funktion c_n^k. Sie dürfen die erweiterte Komposition und das erweiterte rekursive Definitionsschema benützen. LOOP- und WHILE-Programme sind nicht erlaubt.

Lösung

1. Offensichtlich primitiv-rekursiv, weil f eine Konstante ist. Es gilt

   $$\begin{aligned}(\mu f)(n) &= \min\{m \in \mathbb{N}_0\,;\, m^2 \dot{-} n = 0\} \\ &= \min\{m \in \mathbb{N}_0\,;\, m^2 \leq n\} \\ &= 0\,.\end{aligned}$$

2. Sei $k(n) = p(f(n), f(n+1))$.

   Dann gilt

   $$\begin{aligned} k(0) &= p(1,2)\,, \\ k(n+1) &= p(\,c_2(k(n)),\, g(n+2) \cdot c_2(k(n)) \cdot c_1(k(n))\,)\,. \end{aligned}$$

   Mithin ist k μ-rekursiv.

   Wegen $f(n) = c_1(k(n))$ ist damit auch f μ-rekursiv.

Hausaufgabe 3

Sei $L \subseteq \Sigma^*$ und $\prec$ die lexikographische Ordnung auf Wörtern über Σ, und $f : \mathbb{N}_0 \to \Sigma^*$ eine totale und berechenbare Funktion, so dass $L = f(\mathbb{N}_0) = \{f(n) \mid n \in \mathbb{N}_0\}$.
Zeigen Sie: Wenn f monoton ist (d.h. $m < n \implies f(m) \prec f(n)$), dann ist L entscheidbar.

Lösung

Sei $n = |\Sigma|$. Für $k \in \mathbb{N}_0$ gibt es stets weniger als $(n+1)^k$ Wörter w mit $|w| \leq k$.
Wir definieren $L_k = \{f(i) \mid i < (n+1)^k\}$. Für jedes Wort $x \in \Sigma^*$ gilt nun $x \in L$ genau dann, wenn $x \in L_{|x|+1}$. Dieses Prädikat ist offensichtlich entscheidbar.

Hausaufgabe 4

Wir betrachten die in der Vorlesung beschriebene Kodierung von Turingmaschinen durch Wörter über $\Sigma^* = \{0,1\}^*$. Für ein $w \in \Sigma^*$ beschreibt $\varphi_w : \Sigma^* \to \Sigma^*$ dann die Funktion, die durch die Turingmaschine M_w berechnet wird. Finden Sie informelle Beschreibungen für die folgenden Mengen:

1. $C = \{w \in \Sigma^* \,;\, \varphi_w(\epsilon) = w\}$.
2. $D = \{(u,v,w) \in \Sigma^* \times \Sigma^* \times \Sigma^* \,;\, \varphi_u(w) = \varphi_v(w)\}$.

Lösung

1. Die Menge aller (Codes der) Turingmaschinen, die ihren eigenen Code auf das leere Band schreiben. Das entspricht den sogenannten Quines (nach dem Logiker und Philosophen Willard V. O. Quine), also den Programmen, die bei leerer Eingabe ihren eigenen Quellcode ausgeben.
2. Ein Tripel (u,v,w) liegt genau dann in D, wenn die Turingmaschinen M_u und M_v bei der Eingabe w die gleiche Ausgabe liefern. Insbesondere muss auch $M_u[w]\downarrow \iff M_v[w]\downarrow$ gelten.

Hausaufgabe 5

Sei $\Sigma = \{0,1\}$.

1. Wir betrachten das spezielle Halteproblem $K = \{w \in \Sigma^* \,;\, M_w[w]\downarrow\}$ und das Halteproblem auf leerem Band $H_0 = \{w \in \Sigma^* \,;\, M_w[\epsilon]\downarrow\}$.

   Zeigen Sie durch hinreichend genaue Spezifikation und Begründung einer Reduktionsabbildung (wie in den entsprechenden Beweisen der Vorlesung), dass H_0 reduzierbar ist auf K, d.h. $H_0 \leq K$.

2. Zeigen Sie, dass die Menge $R = \{w \in \Sigma^* \,;\, \varphi_w(0) = \perp\}$ unentscheidbar ist.
   Dabei sei φ_w diejenige (partielle) Funktion $\varphi_w : \mathbb{N}_0 \to \mathbb{N}_0$, die von der Turingmaschine M_w berechnet wird.

Lösung

1. Jedem w wird $f(w)$ zugeordnet, so dass die Maschine $M_{f(w)}$ jede Eingabe u löscht und dann auf das leere Band die Maschine M_w anwendet.

   $M_{f(w)}$ hält genau dann auf $f(w)$, wenn M_w auf dem leeren Band hält

2. Satz von Rice mit $F = \{\varphi_w \mid \varphi_w(0) = \perp\}$.

   Es gilt $F \neq \emptyset$ und die berechenbare Konstante 1 liegt nicht in F.

Hausaufgabe 6

Zeigen Sie die Unentscheidbarkeit der folgenden Mengen und wenden Sie zum Beweis Techniken der Reduzierbarkeit eines Problems A auf ein Problem B an.

1. $H_{\Sigma^*} = \{w\,;\, M_w$ hält für mindestens eine Eingabe$\}$.

2. $C = \{w\,;\, M_w$ berechnet die Funktion g mit $g(n) = 0$ für alle n$\}$.

Lösung

Wir gehen davon aus, dass es eine universelle Turingmaschine U gibt, die die Berechnungen jeder Turingmaschine M_w auf deren Eingabe x simulieren kann, und deshalb insbesondere genau dann hält, wenn M_w hält. Außerdem wurde in der Vorlesung bewiesen, dass das Halteproblem H_0 auf leerem Band nicht entscheidbar ist.

1. Wir reduzieren das bekannte Halteproblem H_0 auf das Problem H_{Σ^*} durch Konstruktion einer totalen und berechenbaren Funktion f wie folgt.

   Es sei $w' = f(w)$ der Code einer Turingmaschine $M_{w'}$, die bei Eingabe eines Wortes y folgendes ausführt: Zunächst wird die Turingmaschine M_w bei leerer Eingabe simuliert (beispielsweise auf einem zweiten Band). Falls M_w hält, dann hält auch $M_{w'}$.

   Offenbar gilt nun $w \in H_0 \Leftrightarrow f(w) \in H_{\Sigma^*}$, d.h. f reduziert H_0 auf H_{Σ^*}.

2. Wir verfahren analog zur Lösung der vorhergehenden Aufgabe und reduzieren mit Hilfe einer Funktion f das Problem H_0 auf C:

   Es sei $w' = f(w)$ der Code einer Turingmaschine $M_{w'}$, die bei Eingabe eines Wortes y folgendes ausführt: Zunächst wird die Turingmaschine M_w bei leerer Eingabe simuliert. Falls M_w hält, dann schreibt $M_{w'}$ eine 0 auf das Band und terminiert.

   Offenbar gilt nun $w \in H_0 \Leftrightarrow f(w) \in C$, d.h. f reduziert H_0 auf C.

Hausaufgabe 7

Wahr oder falsch? Begründen Sie im Folgenden Ihre Antworten möglichst knapp!

1. Das PCP $((01, 0), (10, 01), (0, 01))$ besitzt eine Lösung.
2. Wenn f berechenbar ist, dann ist $A_f := \{w \in \Sigma^* \,;\, f(w) \neq \perp\}$ semi-entscheidbar.
3. Für das spezielle Halteproblem $K = \{w \in \{0,1\}^* \,;\, M_w[w] \downarrow\}$ und eine beliebige Sprache A gilt: Wenn $K \cap A$ entscheidbar ist, dann ist A endlich.
4. Das Postsche Korrespondenzproblem ist semi-entscheidbar.
5. Für jede Turingmaschine M ist die Funktion
$$f_M(x) = \begin{cases} 1 & \text{falls } M \text{ auf allen Eingaben hält} \\ 0 & \text{sonst} \end{cases}$$
berechenbar.
6. Wenn f und g primitiv-rekursiv sind, und $f(x) = g(h(x))$ für alle x gilt, dann ist auch h primitiv-rekursiv.
7. Sei $A \subseteq \Sigma^*$. Wenn χ_A total ist, dann ist A entscheidbar.

Lösung

1. (f): Keine Berechnung kann mit einem der Paare enden, weil die letzten Buchstaben der Komponenten nicht gleich sind.
2. (w): $\chi'_{A_f}(x) = \begin{cases} 1 & \text{falls } f(x) \neq \perp \\ \perp & \text{sonst} \end{cases}$ ist berechenbar, da f berechenbar ist.
3. (f): Gegenbeispiel: $A = \overline{K}$. Dann ist $K \cap A = \emptyset$ und deshalb entscheidbar, obwohl A unendlich ist.
4. (w): Systematisch alle Kombinationen aufzählen und prüfen!
5. (w): Für jede Turingmaschine M ist f_M eine konstante Funktion und als solche natürlich berechenbar.
6. (f): Gegenbeispiel: $f(x) = 0$, $g(x) = 0$, und h beliebig und nicht PR.
7. (f): χ_A ist nach Definition immer total. Entscheidend ist, dass χ_A berechenbar ist.

Zusatzaufgabe 1 (Zur Wiederholung)

Die folgende Sprache $L \subseteq \{a, b, c\}^*$ ist nicht vom Typ 2, d.h., sie ist nicht kontextfrei:

$$L = \{a^i b^j c^k \,;\, i, j, k \in \mathbb{N},\, 0 < i < j < k\}\,.$$

1. Stellen Sie L als Durchschnitt kontextfreier Sprachen L_1 und L_2 dar.

   Zeigen Sie die Kontextfreiheit für die von Ihnen gewählten Sprachen L_1 und L_2.

2. Geben Sie eine monotone Grammatik G an, die L erzeugt.

   Hinweis: Monotone Grammatiken haben den Vorteil, dass man mit Produktionen $AB \to BA$ Zeichen in eine Richtung sortieren kann. Solche Produktionen kann man dann durch eine Kette von „kontextsensitiven“ monotonen Produktionen $\alpha A \beta \to \alpha\gamma\beta$ simulieren (siehe TA 2 von Blatt 1).

Lösung

1. Es gilt $L = L_1 \cap L_2$ mit den Sprachen

   $L_1 = \{a^i b^j c^k \,;\, i, j, k \in \mathbb{N}, 0 < i < j\}$ und $L_2 = \{a^i b^j c^k \,;\, i, j, k \in \mathbb{N}, 0 < j < k\}$, die von den folgenden kontextfreien Grammatiken erzeugt werden.

   $$G_1: \begin{array}{lcl} S & \to & TXY \mid TX\,, \\ T & \to & aTb \mid ab\,, \\ X & \to & bX \mid b\,, \\ Y & \to & cY \mid c\,. \end{array} \qquad G_2: \begin{array}{lcl} S & \to & XTY \mid TY\,, \\ T & \to & bTc \mid bc\,, \\ X & \to & aX \mid a\,, \\ Y & \to & cY \mid c\,. \end{array}$$

2. Die folgende Lösung verfolgt den Gedanken, zunächst Satzformen zu produzieren, in denen die Anzahlen i, j, k der Vorkommen von a bzw. b bzw. c korrekt sind, und anschließend die Buchstaben c als Variable C nach rechts zu sortieren. Im Kontext des rechtsständigen Terminalzeichens c kann dann jede Variable C in c umgewandelt werden.

   $$\begin{array}{lcl} S & \to & aS'BBCCc \mid aBBCCc\,, \\ S' & \to & aS'BC \mid T\,, \\ T & \to & BCT \mid BY\,, \\ Y & \to & CY \mid C\,, \end{array} \qquad \begin{array}{lcl} B & \to & b\,, \\ Cc & \to & cc\,. \\ CB & \to & BC\,. \end{array}$$

   Die letzte Regel ist nicht kontextsensitiv. Wir ersetzen diese Regel durch eine Kette von kontextsensitiven Regeln:

   $$CB \to CZ\,, \quad CZ \to BZ\,, \quad BZ \to BC\,.$$

Zusatzaufgabe 2 (Zur Wiederholung)

1. Sei Σ ein Alphabet mit $\# \in \Sigma$. Geben Sie eine deterministische Turingmaschine B an, die die Menge $\{v \in \Sigma^* \,;\, \exists w \in (\Sigma \setminus \{\#\})^*.\ v = \#w\}$ akzeptiert.

2. Wir nennen eine Turingmaschine mit Eingabealphabet Σ und $\# \in \Sigma$ links-markiert, wenn sie sich auf Eingaben $\#w$ mit $w \in (\Sigma \setminus \{\#\})^*$ wie folgt verhält: Nach jedem Berechnungsschritt enthält das Band ein Wort lu mit $u \in (\Gamma \setminus \{\#\})^*$ und $l \in \{\#, \#\#\}$. Links und rechts von lu sei das Band mit Leerzeichen $\Box$ angefüllt.

   Sei $T = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$ mit $\# \notin \Gamma$ (und damit auch $\# \notin \Sigma$) eine deterministische Turingmaschine. Konstruieren Sie eine links-markierte Turingmaschine $T_\#$, so dass für die akzeptierten Sprachen gilt:

   $$L(T_\#) = \{\#w \,;\, w \in L(T)\}\,.$$

   Erläutern Sie Ihre Konstruktion!

   Hinweis: Beachten Sie, dass T an den Wortgrenzen ein Leerzeichen $\Box$ erwartet.

3. Modifizieren Sie Ihre Konstruktion in Punkt 2 derart, dass für die Zustandsmengen Q von T bzw. $Q_\#$ von $T_\#$ jedenfalls $|Q_\#| \le |Q| + 10$ gilt.

   Hinweis: Im Gegensatz zu den Zustandsmengen ist Γ beliebig erweiterbar.

Lösung

1. Seien $B = (Q, \Sigma, \Sigma \cup \{\Box\}, \delta, q_0, \Box, \{q_e\})$ mit $Q = \{q_0, q_1, q_e\}$ und für alle $x \in \Sigma \setminus \{\#\}$:

   $$\delta(q_0, \#) = (q_1, \#, R)\,, \quad \delta(q_1, x) = (q_1, x, R)\,, \quad \delta(q_1, \Box) = (q_e, \Box, N)\,.$$

2. *Erläuterung*:
   Sei $T_\# = (Q_\#, \Sigma, \Gamma_\#, \delta_\#, q_\#, \Box, F)$ mit $Q_\# = Q \cup \{s_q \mid q \in Q\} \cup \{q_\#\}$.
   Dabei bezeichne s_q eine Kopie von q.

   Für ein Eingabewort $\#w$ mit $w \in \Sigma^*$ steht beim Start der Kopf von $T_\#$ auf $\#$. Akzeptiert wird genau dann, wenn w von T akzeptiert wird.

   Zunächst wird der Kopf auf den ersten Buchstaben von w gesetzt. Falls $w = \epsilon$, dann steht der Kopf auf $\Box$ rechts neben $\#$.

   Nun wird T gestartet. Wenn T auf $\#$ trifft, dann wird $\#$ nach links versetzt und dabei aber der momentane Zustand q in s_q gespeichert! Dann wird ein $\Box$ eingefügt und T auf dem eingefügten $\Box$ mit dem Zustand q wieder gestartet.

   Konstruktion von $\delta_\#$ durch Erweiterung von δ für alle $q \in Q$:

   $$\begin{aligned} \delta_\#(q_\#, \#) &= (q_0, \#, R)\,, \quad & \delta_\#(q, \#) &= (s_q, \#, L)\,, \\ \delta_\#(s_q, \Box) &= (s_q, \#, R)\,, \quad & \delta_\#(s_q, \#) &= (q, \Box, N)\,. \end{aligned}$$

3. Die Speicherung des Zustands bei der Versetzung von # erfolgt nun mit neuen Bandzeichen aus $\{\gamma_q \mid q \in Q\}$. Allerdings erfordert die Einhaltung der Berechnungsbedingungen die Verwendung von Doppelkreuzen als Begrenzung.

$$\delta_\#(q_\#, \#) = (q_{\#\#}, \#, L)\,, \quad \delta_\#(q_{\#\#}, \Box) = (q_{\#\#}, \#, R)\,, \quad \delta_\#(q_{\#\#}, \#) = (q_0, \#, R)\,.$$

Man modifiziert dann für alle $q \in Q$ und neuen Zuständen q_1, q_2

$$\begin{aligned} \delta_\#(q, \#) &= (q_1, \gamma_q, L)\,, \quad \delta_\#(q_1, \#) = (q_1, \#, L)\,, \quad \delta_\#(q_1, \Box) = (q_2, \#, R)\,, \\ \delta_\#(q_2, \#) &= (q_2, \#, R)\,, \quad \delta_\#(q_2, \gamma_q) = (q, \Box, N)\,. \end{aligned}$$

Zusatzaufgabe 3 (Zur Wiederholung)

Sei $\Sigma = \{*, \#\}$. Wir kodieren ganze Zahlen $n \in \mathbb{N}_0$ als Folge $** \ldots *$ der Länge n, d. h. $|** \ldots *| = n$, und stellen Paare $(x, y) \in \{*\}^* \times \{*\}^*$ als Wort $x\#y \in \Sigma^*$ dar.
Wir betrachten für $x, y, z \in \{*\}^*$ die Addition $|z| = |x| + |y|$.

1. Definieren Sie durch Angabe der Übergangsfunktion δ eine linear beschränkte Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$, die für $x, y, z \in \{*\}^*$ die Addition $|z| = |x| + |y|$ wie folgt durchführt:

   Startkonfiguration: $(\epsilon, q_0, x\#y)$. Endkonfiguration: (ϵ, q_e, z), mit $q_e \in F$.

   Es gilt: $(\epsilon, q_0, x\#y) \underset{M}{\rightarrow}^* (\epsilon, q_e, z)$.

   Beschreiben Sie kurz die Konstruktionsidee für Ihre Maschine.

2. Seien c_1, c_2 die Umkehrfunktionen einer Paarfunktion $p : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$. Dann ist $plus : \mathbb{N}_0 \to \mathbb{N}_0$ mit $plus(n) = c_1(n) + c_2(n)$ die Kodierung der Addition nichtnegativer ganzer Zahlen, d.h., $x + y = plus(p(x, y))$ für alle $x, y \in \mathbb{N}_0$.

   Zeigen Sie die Unentscheidbarkeit der folgenden Menge P:

   $$P = \{w \in \{0,1\}^*\,;\ \text{die von } M_w \text{ berechnete Funktion ist gleich } plus\}\,.$$

3. Sei $H_0 = \{w \in \{0,1\}^*\,;\ M_w \text{ hält auf leerem Band}\}$ das Halteproblem auf leerem Band. Zeigen Sie durch informelle Spezifikation einer Reduktionsabbildung f (wie in entsprechenden Beweisen der Vorlesung), dass H_0 reduzierbar ist auf P, i. Z. $H_0 \leq P$.

Lösung

1. Idee: Falls $x \neq \epsilon$, dann wird das erste Zeichen $*$ in x gelöscht und das Zeichen # durch $*$ ersetzt. Falls $x = \epsilon$, dann wird nur # gelöscht. Schließlich wird der Kopf nach vorne positioniert.

   Seien $Q = \{q_0, q_1, q_2, q_e\}$, $\Delta = \{*, \#, \Box\}$ und $F = \{q_e\}$.

   $$\begin{aligned} \delta(q_0, *) &= (q_1, \Box, R)\,, & \delta(q_0, \#) &= (q_e, \Box, R)\,, \\ \delta(q_1, *) &= (q_1, *, R)\,, & \delta(q_1, \#) &= (q_2, *, L)\,, \\ \delta(q_2, *) &= (q_2, *, L)\,, & \delta(q_2, \Box) &= (q_e, \Box, R)\,. \end{aligned}$$

2. Sei $S = \{plus\}$ die einelementige Menge von berechenbaren Funktionen $\mathbb{N}_0 \to \mathbb{N}_0$, mit dem berechenbaren Element $plus$.

   Es gilt $S \neq \emptyset$ und S ungleich der Menge aller Funktionen. Damit ist der Satz von Rice anwendbar, der beweist, dass P unentscheidbar ist.

3. Sei w_p der Code einer Turingmaschine M_{w_p}, die $plus$ berechnet.

   Für alle $w \in \{0,1\}^*$ sei $f(w)$ der Code einer Turingmaschine M, die wie folgt definiert ist:

   M simuliert eine 2-Band-Turingmaschine, die die Eingabe von M auf Band 1 schreibt und anschließend auf Band 2 die Turingmaschine M_w auf leerem Band ausführt.

   Falls M_w hält, dann wird auf Band 1 die Eingabe von M mit M_{w_p} ausgeführt und das Ergebnis auf das Band von M geschrieben.

   Offenbar ist f eine totale und berechenbare Funktion, so dass $f(H_0) \subseteq P$ und $f(\overline{H_0}) \subseteq \overline{P}$ gelten.

Zusatzaufgabe 4 (Zur Wiederholung)

Wahr oder falsch? Begründen Sie Ihre Antwort!

1. Falls eine Grammatik Chomsky-Normalform besitzt, dann enthält sie keine nutzlosen Variablen.

2. Falls $L \subseteq \Sigma^*$ deterministisch kontextfrei ist, dann gibt es eine $LR(k)$ Grammatik, die das Komplement $\overline{L} = \Sigma^* \setminus L$ erzeugt.

3. Sei $\Sigma = \{0,1\}$. Das Komplement $\overline{H_s} = \Sigma^* \setminus H_s$ des speziellen Halteproblems H_s ist eine Typ-0-Sprache. (H_s wurde in Übungen auch als K bezeichnet.)

4. Die Menge $\{w \in \{0,1\}^*\,;\, \varphi_w \text{ ist } \mu\text{-rekursiv}\}$ ist entscheidbar. Dabei ist φ_w die von der Turingmaschine M_w berechnete Funktion.

Lösung

1. Falsch! Es können neue, nutzlose Variable mit entsprechenden Produktionen hinzugefügt werden.

2. Wahr! Die Klasse der deterministisch kontextfreien Sprachen ist abgeschlossen für Komplementbildung. Zu jeder DCFL gibt es eine erzeugende $LR(k)$ Grammatik.

3. Falsch! $\overline{H_s}$ ist nicht semi-entscheidbar, weil H_s nicht entscheidbar, aber semi-entscheidbar ist.

4. Wahr! Für alle w ist φ_w berechenbar und folglich μ-rekursiv.

Zusatzaufgabe 5 (Zur Wiederholung)

Wahr oder falsch? Begründen Sie Ihre Antwort!

1. Sei $L \subseteq \Sigma^*$ eine nichtleere reguläre Sprache. Dann enthält der Rechtsquotient L/L das leere Wort ϵ.
2. Seien $A, B \subseteq \Sigma^*$. Falls A auf B (effektiv) reduzierbar (wie in der Vorlesung definiert) und B regulär ist, dann ist auch A regulär.
3. Eine Sprache ist genau dann rekursiv aufzählbar (r.a., semi-entscheidbar), wenn sie vom Typ 0 ist.
4. Es ist entscheidbar, ob eine Turingmaschine M die Ackermann-Funktion berechnet.
5. Das allgemeine Halteproblem H ist semi-entscheidbar.

Lösung

1. Wahr! Sei $x \in L$. $x = \epsilon x \implies \epsilon \in L/L$.
2. Falsch! Seien $A = \{a^n b^n\,;\, n \in \mathbb{N}\}$, $B = \{\epsilon\}$ und $f(w) = \epsilon$ für $w \in A$, und $f(w) = a \in \Sigma$ sonst.
3. Wahr! Nach Satz der Vorlesung sind die Typ-0-Sprachen der Chomsky Hierarchie genau die von Turingmaschinen M akzeptierten Sprachen $L(M)$. Jede Sprache $L = L(M)$ ist semi-entscheidbar, weil deren „semi-charakteristische" Funktion χ'_L berechenbar ist durch eine Turingmaschine M', die 1 ausgibt, falls M hält, und ansonsten nicht hält.
4. Falsch! Mit Satz von Rice.
5. Das allgemeine Halteproblem $H = \{\langle x, w\rangle_c\,;\, M_w,\ \text{angesetzt auf } x,\ \text{hält}\}$ fasst man als Teilmenge von $\mathbb{N}_0$ auf. Um dies zu verdeutlichen haben wir das Paar $\langle x, w\rangle$ für Zwecke dieser Aufgabe mit einem Index c ausgestattet. Das c soll eine Abbildung der Menge von Paaren $\langle x, w\rangle$ in $\mathbb{N}_0$ bedeuten. Wir hatten in der Vorlesung Paare durch die Paarfunktionen aufgezählt. Damit gilt also $H \subseteq \mathbb{N}_0$. Man kann alternativ auch mit $H \subseteq \{0,1\}^*$ argumentieren.

   Wir müssen nun die Existenz einer Turingmaschine T mit $L(T) = H$ nachweisen oder zumindest erklären.

   Wir beschreiben T.

   Bei Eingabe einer Zahl $n \in \mathbb{N}_0$ berechnet T die zugehörige Eingabe x, berechnet die zweite Komponente w, berechnet die zugehörige Turingmaschine M_w und simuliert die Berechnung von M_w bei Eingabe von x. T hält genau dann, wenn die simulierte Turingmaschine hält, d. h. wenn $n \in H$ gilt.

Zusatzaufgabe 6 (Zur Wiederholung)

Sei $\Sigma = \{b, \#, \mathsf{x}\}$; dabei heiße b *Buchstabe*, $\#$ *Zeilenendezeichen* und x *Löschzeichen*. Jedes Wort $w\# \in \Sigma^*$ heiße eine *Seite* über Σ. Ein Wort $w\# \in \Sigma^*$ heiße *Zeile*, falls $\#$ nicht in w vorkommt. Falls in w kein Buchstabe b vorkommt, dann heißt $w\#$ *Leerzeile*. Sei $\#_b(w)$ die Anzahl der b, die in w vorkommen. Ein Wort $s = z_0z_1 \dots z_m$ mit Zeilen z_i und $m \geq 0$ heiße *formatierte Seite*, falls $\#_b(z_0) = \#_b(z_i)$ für alle $i \leq m$ gilt.
Beispiel: das Wort $b\mathsf{x}b\#\mathsf{xx}b\mathsf{x}b\mathsf{x}\#bb\#$ ist eine formatierte Seite mit drei Zeilen. Die formatierte Seite $\mathsf{x}\#\#$ besteht aus zwei Leerzeilen.

1. Definieren Sie durch Angabe der Übergangsfunktion δ und der Menge F der Endzustände eine Turingmaschine $M_l = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$, die für alle Eingaben $e \in \Sigma^*$ hält, kein Zeichen auf dem Band überschreibt und die Menge S_l aller (formatierten) Seiten akzeptiert, die aus lauter Leerzeilen bestehen. Vor dem Start stehe der Kopf über dem ersten Zeichen und nach dem Halt stehe der Kopf über dem letzten Zeichen einer nicht leeren Eingabe.

   Geben Sie jeweils die Idee der Bedeutung der Zustände an.

2. Wir nehmen an, dass M_b eine Turingmaschine sei, die für alle Eingaben $e \in \Sigma^*$ hält, kein Zeichen auf dem Band überschreibt und die Menge S_b aller Seiten akzeptiert, die keine Leerzeilen enthalten. Vor dem Start stehe der Kopf über dem ersten Zeichen und nach dem Halt stehe der Kopf über dem letzten Zeichen einer nicht leeren Eingabe.

   Die Menge S_f der formatierten Seiten über Σ ist keine Sprache vom Typ 2, d.h. S_f ist nicht kontextfrei. Beschreiben Sie eine auf M_l und M_b gestützte Konstruktionsidee für eine linear beschränkte Turingmaschine M_f, die S_f akzeptiert.

Lösung

1. Seien $Q = \{q_0, q_1, r, s, f\}$, $\Gamma = \{b, \#, \mathsf{x}, \Box\}$ und $F = \{f\}$. Im Zustand s stoppt die Maschine, ohne zu akzeptieren. Im Zustand r wird nach rechts gelaufen und in s gestoppt.

$$\begin{array}{lcllcl}
\delta(q_0, \Box) & = & (s, \Box, L)\,, & \delta(q_0, b) & = & (r, b, R)\,, \\
\delta(q_0, \mathsf{x}) & = & (q_0, \mathsf{x}, R)\,, & \delta(q_0, \#) & = & (q_1, \#, R)\,, \\
\delta(q_1, \mathsf{x}) & = & (q_0, \mathsf{x}, R)\,, & \delta(q_1, \#) & = & (q_1, \#, R)\,, \\
\delta(q_1, b) & = & (r, b, R)\,. & \delta(q_1, \Box) & = & (f, \Box, L)\,, \\
\delta(r, *) & = & (r, *, R)\,, & \delta(r, \Box) & = & (s, \Box, L)\,, \quad * \in \{b, \#, \mathsf{x}\}\,.
\end{array}$$

2. Konstruktionsidee für M_f:
   1. Es wird mit M_l geprüft, ob die Eingabe nur aus Leerzeilen besteht. Wenn ja, dann hält M_f und akzeptiert!
   2. Wenn nein, dann wird der Kopf an den Anfang des Bandes gefahren und M_b ausgeführt. Falls die Seite eine Leerzeile enthält, dann hält M_f und akzeptiert nicht! Falls die Seite keine Leerzeile enthält, dann wird am Seitenende eine Maschine D gestartet, die an den Seitenanfang läuft und dabei in jeder Zeile ein b löscht, d.h. durch das Löschzeichen x ersetzt.
   3. Fortsetzung mit 1.