Technische Universität München
Fakultät für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Werner Meixner, Dr. Alexander Krauss

Sommersemester 2010
Lösungsblatt 9
2. Juli 2010

Einführung in die Theoretische Informatik

Hausaufgabe 1 (5 Punkte)

Seien $\Sigma = \{a, b\}$ und $L = \{w \in \Sigma^* \mid w = uav, u, v \in \Sigma^*, |u|_a = |v|_a, |u|_b = |v|_b\}$.
Geben Sie einen 2-Kellerautomaten an (Definition siehe Tutoraufgabe 1), der L akzeptiert.

Lösungsvorschlag

Entsprechend der Definition in Tutoraufgabe 1 setzen wir
$K = (\{q_0, q_1, q_f\}, \{a, b\}, \{a, b, Z_0, Z_0'\}, \delta, q_0, Z_0, Z_0', \{q_f\})$. Wir benutzen je einen Keller für die Zählung von a's bzw. b's.
Die Bezeichnungen x, y bedeuten im Folgenden Variable für Kellerzeichen. Das Elementzeichen $\ni$ schreiben wir zwecks besserer Lesbarkeit als Relationspfeil $\to$. Wir definieren für alle $x, y \in \Gamma$.

$$\begin{aligned}
\delta(q_0, a, x, y) &\to (q_0, ax, y), & \delta(q_0, a, x, y) &\to (q_0, bx, y),\\
\delta(q_0, a, x, y) &\to (q_1, x, y), & \delta(q_0, \epsilon, Z, Z') &\to (q_f, \epsilon, \epsilon),\\
\delta(q_1, a, a, y) &\to (q_1, \epsilon, y), & \delta(q_1, b, x, b) &\to (q_1, x, \epsilon),\\
\delta(q_1, \epsilon, Z, Z') &\to (q_f, \epsilon, \epsilon).
\end{aligned}$$

Bemerkung: Die Sprache L ist nicht kontextfrei. Es existiert also kein Kellerautomat (d. h. mit einem einzigen Keller), der L akzeptiert. Man kann aber zeigen, dass L kontextsensitiv ist und damit von einem linear beschränkten Automaten akzeptiert wird.

Hausaufgabe 2 (5 Punkte)

Seien $\Sigma = \{a_1, a_2, \ldots, a_n\}$ ein beliebiges n-elementiges Alphabet und $\Sigma' = \Sigma \cup \{\#\}$.
Geben Sie eine Turingmaschine $M = (Q, \Sigma', \Gamma, \delta, q_0, \square, F)$ mit höchstens 5 Zuständen an, die bei leerer Eingabe das Alphabet Σ in der Form $\#a_1\#a_2 \ldots \#a_n$ auf das Band schreibt und mit dem Kopf auf dem letzten, rechtsstehenden Zeichen der Ausgabe anhält.

Lösungsvorschlag

Sei $M = (\{q_0, q_L, q_A, q_f\}, \Sigma', \Sigma' \cup \{\square\}, \delta, q_0, \square, \{q_f\})$.

Übergang		Bereich	Kommentar
$\delta(q_0, \square)$	$\to (q_L, \square, L)$		Übergang in Linkslauf an Wortanfang.
$\delta(q_0, a_i)$	$\to (q_0, a_{i+1}, R)$	$i < n$	„Shift“ des Alphabets.
$\delta(q_0, \#)$	$\to (q_0, \#, R)$		„Shift“ des Alphabets.
$\delta(q_L, a_n)$	$\to (q_f, a_n, R)$		Ende.
$\delta(q_L, a_i)$	$\to (q_L, a_i, L)$	$i < n$	nach links gehen
$\delta(q_L, \#)$	$\to (q_L, \#, L)$		nach links gehen
$\delta(q_L, \square)$	$\to (q_A, \#, L)$		Schreiben am Wortanfang.
$\delta(q_A, \square)$	$\to (q_0, a_1, R)$		Schreiben von a_1 und Start.

Hausaufgabe 3 (5 Punkte)

Eine 1-Band Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ heißt linear beschränkter Automat, wenn sie für keine Eingabe ein Leerzeichen $\square$ überschreibt.
Sei $\Sigma = \{0, 1\}$. Geben Sie für die kontextsensitive Sprache $L = \{ww \mid w \in \Sigma^*\}$ einen linear beschränkten Automaten M an, der L akzeptiert.

Lösungsvorschlag

Für $v \in \Sigma^*$ sucht man zunächst die Mitte und prüft anschließend buchstabenweise die Gleichheit der entsprechenden Teilwörter v_1, v_2 von $v = v_1 v_2$. Die Mitte eines Wortes v kann man durch wiederholte, unterschiedliche Markierung des ersten bzw. letzten Buchstaben von v suchen.
Für Markierungen des linken bzw. rechten Teilwortes v_1 bzw. v_2 benützen wird die Alphabete $\Sigma_a = \{(x, 1)\,;\, x \in \Sigma\}$ und $\Sigma_A = \{(x, 2)\,;\, x \in \Sigma\}$, wobei beide Alphabete als Kopien von Σ aufgefasst werden können. Wir schreiben für Elemente $x \in \Sigma$ das entsprechende Element in Σ_a bzw. Σ_A als x_a bzw. x_A.
Als Bandalphabet des zu konstruierenden LBA definieren wir $\Gamma = \Sigma \cup \Sigma_a \cup \Sigma_A \cup \{X, \square\}$ mit dem Hilfszeichen X.

Der Anfangszustand sei q_0, der Endzustand q_f. Für jedes Symbol $x \in \Sigma$ definieren wir einen Zustand q_x. Die Zustände q_x werden die Aufgabe übernehmen, sich den Buchstaben x zu „merken“. Die Übergangsfunktion sei in der folgenden Tabelle gegeben für alle $x \in \Sigma$.

Übergang			Kommentar
$\delta(q_0, \square)$	$\to$	$(q_f, \square, N)$	das leere Eingabewort wird akzeptiert
$\delta(q_0, x)$	$\to$	(q_0', x, N)	ein nichtleeres Wort wird q_0' zur Verarbeitung gegeben
$\delta(q_0', x)$	$\to$	(q_1, x_a, R)	Markierung des ersten Buchstaben des linken Teilwortes
$\delta(q_1, x)$	$\to$	(q_1, x, R)	die Σ-Eingabe wird nach rechts überlaufen
$\delta(q_1, \square)$	$\to$	$(q_2, \square, L)$	Stopp erster Rechtsdurchlauf bis zum rechten Rand
$\delta(q_1, x_A)$	$\to$	(q_2, x_A, L)	Stopp Rechtsdurchlauf bis zum rechten Teilwort
$\delta(q_2, x)$	$\to$	(q_3, x_A, L)	Markierung x durch x_A und Übergang zum Rücklauf
$\delta(q_3, x)$	$\to$	(q_4, x, L)	für $x \in \Sigma$ Übergang in Linkslauf
$\delta(q_3, x_a)$	$\to$	(q_5, x_a, N)	Stopp Rücklauf und Beginn der 2.Phase
$\delta(q_4, x)$	$\to$	(q_4, x, L)	Linkslauf
$\delta(q_4, x_a)$	$\to$	(q_0', x_a, R)	Schleife

Zustand q_5 wird nur dann erreicht, wenn das Eingabewort v in gleichlange Wörter v_1, v_2 zerlegbar ist. v_1 und v_2 sind dann markiert. Der Schreib-/Lesekopf befindet sich auf dem letzten, entsprechend markierten Buchstaben von v_1.

Übergang			Kommentar
$\delta(q_5, x_a)$	$\to$	(q_x, X, R)	Speichern von x und Markierung der Stelle
$\delta(q_x, X)$	$\to$	(q_x, X, R)	
$\delta(q_x, y_A)$	$\to$	(q_x, y_A, R)	Überfahren des rechten Teilworts
$\delta(q_x, \square)$	$\to$	$(q_x', \square, L)$	umkehren und x merken
$\delta(q_x', x_A)$	$\to$	$(q_7, \square, L)$	
$\delta(q_7, z)$	$\to$	(q_7, z, L)	mit $z = x_A$ oder $z = X$
$\delta(q_7, x_a)$	$\to$	(q_5, x_a, N)	
$\delta(q_7, \square)$	$\to$	$(q_f, \square, N)$	erfolgreiches Ende

Hausaufgabe 4 (5 Punkte)

Zeigen Sie, dass man **case**-Anweisungen der folgenden Form für festes n durch LOOP-Programme simulieren kann:

$$\textbf{case } x_i \textbf{ do } \; 0\!:\!P_0;\; 1\!:\!P_1;\; \ldots\; n\!:\!P_n \; \textbf{ od}$$

Falls $0 \leq x_i \leq n$ gilt, dann wird das LOOP-Programm P_{x_i} ausgeführt und die **case**-Anweisung beendet. Ist $x_i > n$, so wird die **case**-Anweisung übersprungen. Keines der LOOP-Programme P_j enthalte die Variable x_i.
Verwenden Sie nur Zuweisungen der Form $x_k := x_j \pm c$, den LOOP-Befehl und Initialisierungen der Form $x_k := c$.

Lösungsvorschlag

Offenbar ist die **case**-Anweisung äquivalent zu der wiederholten **ifthen**-Anweisung

$$\begin{array}{ll} \textbf{if } x_i = 0 & \textbf{then } P_0 \\ \textbf{if } x_i = 1 & \textbf{then } P_1 \\ \quad \vdots & \\ \textbf{if } x_i = n & \textbf{then } P_n \end{array}$$

Für jede Zeile benötigen wir das folgende LOOP-Programm, zunächst in Abhängigkeit von einer Schemavariablen j. Eine vorausgehende, notwendige Nullstellung der Variablen $x_0, \ldots, x_4$ notieren wir nicht eigens.

Programmbezeichnung Q_j:

$$\begin{array}{ll} x_0 := x_i + 0; & \\ x_1 := x_1 + j; & \\ x_2 := x_0 + 0; & LOOP\ x_1\ DO\ x_2 := x_2 - 1\ END; \\ x_3 := x_1 + 0; & LOOP\ x_0\ DO\ x_3 := x_3 - 1\ END; \\ x_4 := x_4 + 1; & LOOP\ x_3\ DO\ x_4 := x_4 - 1\ END; \\ & LOOP\ x_4\ DO\ P_j \end{array}$$

Nun ergibt sich das gesuchte Programm als

$$Q_0;\; Q_1;\; \ldots; Q_n$$

Vorbereitung 1

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

$$f(n+1) = n \cdot f(n-1) + f(n-2) \quad \forall n \in \mathbb{N}, n \geq 2\,.$$

Geben Sie ein LOOP-Programm an, das den Funktionswert von f für $n \geq 3$ in der Variablen x_3 berechnet.

Lösungsvorschlag

Für gegebenes $n \geq 3$ berechnen wir den Funktionswert $f(n)$ in der Variablen x_3. Dann sollen x_2 bzw. x_1 bzw. x_0 die Werte $f(n-1)$ bzw. $f(n-2)$ bzw. $f(n-3)$ enthalten. Es enthalte x_4 das Argument n als gegebene Konstante. P sei ein LOOP-Programm, das die Multiplikation $x_6 = x_5 \cdot x_2$ berechnet. Wir schreiben wie in der Vorlesung $x_6 := x_5 * x_2$ für P.

$x_4 := n;$
$x_0 := 1;\ x_1 := 1;\ x_2 := 1;$
$x_4 := x_4 - 2;$
$x_5 := 2;$
$LOOP\ x_4\ DO$
  $x_6 := x_5 * x_2;$
  $x_3 := x_6 + x_0;$
  $x_0 := x_1;\ x_1 := x_2;\ \ x_2 := x_3;$
  $x_5 := x_5 + 1$
END

Vorbereitung 2

Geben Sie eine auf den in der Vorlesung definierten Regeln für primitiv rekursive Funktionen basierende Definition an für $\mathit{fak}(n) = n!\,.$

Lösungsvorschlag

Die Konstante 1 und die Multiplikation $\mathit{mult}(x, y)$ ist nach Vorlesung PR. Wir benutzen außerdem die erweiterte Komposition von primitiv rekursiven Funktionen, die es erlaubt in

$$h(x, y) = \mathit{mult}(\pi_1^2(x, y), s(\pi_2^2(x, y)))$$

die Projektionen zu entfernen und dadurch wie folgt abzukürzen.

$$h(x, y) = \mathit{mult}(x, s(y))\,.$$

Nun folgt die primitive Rekursivität der Fakultätsfunktion aus

$$\begin{aligned} \mathit{fak}(0) &= 1 \\ \mathit{fak}(n+1) &= \mathit{mult}(\mathit{fak}(n), s(n)). \end{aligned}$$

Tutoraufgabe 1

Ein 2-Kellerautomat $K = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, Z_0', F)$ ist ein Kellerautomat, der über einen zweiten Keller verfügt. Der zweite Keller wird mit Z_0' initialisiert. Die Übergangsfunktion $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \times \Gamma \to \mathcal{P}_e(Q \times \Gamma^* \times \Gamma^*)$ beschreibt die Vorgehensweise des 2-KA wie folgt ($\mathcal{P}_e$ bezeichnet die Menge aller endlichen Teilmengen): Liest der 2-KA im Zustand q die Eingabe b (auch $b = \epsilon$ ist möglich), sind Z_1, Z_2 die obersten Zeichen der beiden Keller und gilt $(q', \alpha_1, \alpha_2) \in \delta(q, b, Z_1, Z_2)$, dann kann der 2-KA in den Zustand q' übergehen und hierbei Z_1 durch α_1 und Z_2 durch α_2 ersetzen.

Zeigen Sie: Jede 1-Band-Turingmaschine $T = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$ kann durch einen 2-Kellerautomaten $K = (Q', \Sigma, \Gamma', \delta', q_0', Z_0, Z_0', F')$ simuliert werden.

Lösungsvorschlag

Bemerkung: Bei einer Simulation müssen die Berechnungen bzw. Konfigurationsänderungen zweier Machinen einander zugeordnet werden können und die akzeptierten Sprachen müssen gleich sein.

Die Idee für die Simulation von $T = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$ ist, eine Turing-Konfiguration (α, q, β) darzustellen durch die Konfiguration $(q, \epsilon, \alpha^R Z_0, \beta Z_0')$, wobei α^R gleich dem gespiegelten Wort α ist. Konfigurationsübergänge werden im Kellerautomaten stets mit leeren Eingaben ϵ modelliert.
Beim Start von T liegt das Wort x auf dem Band, der Kopf von T befindet sich über dem ersten Element von $x\Box$. Dies bedeutet, dass wir diese Anfangskonfiguration nach dem Start von K erst herstellen müssen. Dazu dient die folgende Konstruktion.

Sei $K = (Q \cup \{q_0', q_1'\}, \Sigma, \Gamma \cup \{Z_0, Z_0'\}, \delta', q_0', Z_0, Z_0', F)$.
Der 2-Kellerautomat legt zuerst die Eingabe auf den linken Keller. Danach wird der Inhalt des linken Kellers auf den rechten Keller gelegt, wobei der linke Keller fast, d. h. bis auf Z_0, geleert wird. Die beiden Keller entsprechen dann dem Teil des Bandes der Turing-Maschine, der links bzw. rechts vom Lesekopf ist.

$$
\begin{aligned}
\delta'(q_0', a, *, *) &= \{(q_0', a*, *)\}\,, && \text{Eingabe auf linken Keller legen.}\\
\delta'(q_0', \epsilon, *, *) &= \{(q_1', *, *)\}\,, && \text{Eingabe fertig.}\\
\delta'(q_1', \epsilon, A, *) &= \{(q_1', \epsilon, A*)\}\,, && A \neq Z_0,\ \text{linken Keller} \to \text{rechten Keller.}\\
\delta'(q_1', \epsilon, Z_0, *) &= \{(q_0, Z_0, *)\}\,.
\end{aligned}
$$

Nun beginnt die eigentliche Simulation der Turingmaschine T, wobei δ' definiert ist für $a \neq Z_0$, $b \neq Z_0'$ durch

$$
\begin{aligned}
\delta'(q, \epsilon, a, b) \quad = \quad & \{(q', ca, \epsilon) \mid (q', c, R) \in \delta(q, b)\}\\
& \cup \{(q', \epsilon, ac) \mid (q', c, L) \in \delta(q, b)\}\\
& \cup \{(q', a, c) \mid (q', c, N) \in \delta(q, b)\}\,,
\end{aligned}
$$

für $a = Z_0, b \neq Z_0'$ durch

$$
\begin{aligned}
\delta'(q, \epsilon, Z_0, b) \quad = \quad & \{(q', cZ_0, \epsilon) \mid (q', c, R) \in \delta(q, b)\}\\
& \cup \{(q', Z_0, \Box c) \mid (q', c, L) \in \delta(q, b)\}\\
& \cup \{(q', Z_0, c) \mid (q', c, N) \in \delta(q, b)\}\,,
\end{aligned}
$$

für $a \neq Z_0, b = Z_0'$ durch

$$\begin{aligned}\delta'(q, \epsilon, a, Z_0') &= \{(q', ca, Z_0') \mid (q', c, R) \in \delta(q, \square)\} \\ &\cup \{(q', \epsilon, acZ_0') \mid (q', c, L) \in \delta(q, \square)\} \\ &\cup \{(q', a, cZ_0') \mid (q', c, N) \in \delta(q, \square)\},\end{aligned}$$

für $a = Z_0, b = Z_0'$ durch

$$\begin{aligned}\delta'(q, \epsilon, Z_0, Z_0') &= \{(q', cZ_0, Z_0') \mid (q', c, R) \in \delta(q, \square)\} \\ &\cup \{(q', Z_0, \square cZ_0') \mid (q', c, L) \in \delta(q, \square)\} \\ &\cup \{(q', Z_0, cZ_0') \mid (q', c, N) \in \delta(q, \square)\}.\end{aligned}$$

Tutoraufgabe 2

Zeigen Sie durch Rückführung auf die Definition, dass die folgenden Funktionen primitiv rekursiv sind.

1. $\mathit{twopow}(n) = 2^n$,
2. $\mathit{tower}(n) = 2^{2^{2^{\cdot^{\cdot^{\cdot^2}}}}}$ (d.h. $2^{(2^{(2^{\cdot^{\cdot^{\cdot}}})})}$, Turm der Höhe n),
3. $\mathit{ifthen}(n, a, b)$ mit

$$\mathit{ifthen}(n, a, b) = \begin{cases} a & n \neq 0, \\ b & n = 0. \end{cases}$$

Lösungsvorschlag

Die Konstante 1 und die Multiplikation $\mathit{mult}(x, y)$ ist nach Vorlesung PR. Wir benutzen außerdem die erweiterte Komposition von primitiv rekursiven Funktionen

1. $$\begin{aligned}\mathit{twopow}(0) &= 1 \\ \mathit{twopow}(n+1) &= \mathit{mult}(\mathit{twopow}(n), 2).\end{aligned}$$
2. $$\begin{aligned}\mathit{tower}(0) &= 1 \\ \mathit{tower}(n+1) &= \mathit{twopow}(\mathit{tower}(n)).\end{aligned}$$
3. $$\begin{aligned}\mathit{ifthen}(0, a, b) &= b \\ \mathit{ifthen}(n+1, a, b) &= a.\end{aligned}$$

Tutoraufgabe 3

Sei $f : \mathbb{N} \to \mathbb{N}$ diejenige Funktion, die für alle $n \in \mathbb{N}, n \neq 0$ durch die Rekursion

$$f(n+1) = f(n) \cdot f(n-1)$$

mit den Startwerten $f(0) = 1$ und $f(1) = 2$ definiert ist.

1. Zeigen Sie, dass f primitiv-rekursiv ist.
2. Zeigen Sie, dass f streng monoton wachsend ist, d. h., es gilt $f(n) < f(n+1)$ für alle $n \in \mathbb{N}$ mit $n \neq 1$.
3. Ist der Wertebereich von f entscheidbar?

Lösungsvorschlag

1. Das folgende LOOP-Programm basiert auf den Variablen $x_0, \ldots, x_5$. In x_0 wird das Ergebnis $f(n)$ ausgegeben, x_1 enthält beim Start das Argument n.

$$
\begin{array}{l}
x_1 := n; \\
x_2 := x_1 - 1; \\
x_3 := 1;\ x_4 := 2; \\
LOOP\ x_2\ DO \\
\quad x_5 := x_4 * x_3; \\
\quad x_3 := x_4;\ x_4 := x_5; \\
END; \\
x_0 := x_5 \\
IF\ x_1 = 0\ THEN\ x_0 := 1\ END; \\
IF\ x_1 = 1\ THEN\ x_0 := 2\ END
\end{array}
$$

   Wenn wir Satz 4.41 der Vorlesung benutzen, dann folgt aus der Existenz eines LOOP-Programms für die Funktion f bereits die primitive Rekursivität von f.

   Wir wollen nun aber einen alternativen Beweis der PR von f geben, der den Beweis des Satzes 4.41 erhellt. Wir stützen uns dabei auf die Definition der primitiven Rekursivität und erweiterten Schemata zur Definition primitiv rekursiver Funktionen.

   Die Schwierigkeit der Anwendung der Schemata der primitiven Rekursion auf die Funktion f besteht darin, dass die Funktionswerte $f(n+1)$ nicht allein von $f(n)$ sondern auch von $f(n-1)$ abhängen. Dieses Problem kann man angehen, in dem man Tupel von natürlichen Zahlen als natürliche Zahl kodiert, etwa so wie das in dem Beweis von Satz 4.41 gemacht wird. Wir simulieren aber nicht das LOOP-Programm, wie in dem Beweis von 4.41, sondern wir extrahieren die wichtigsten Schritte.

   In Tupel- bzw. Vektorschreibweise liest sich die Rekursion wie folgt. Wir suchen eine Funktion f, so dass $(f(0), f(1))^T = (1,2)^T$ und für alle $n \in \mathbb{N}$ die folgende Gleichung gilt.

$$\begin{pmatrix} f(n+1) \\ f(n+2) \end{pmatrix} = \begin{pmatrix} f(n+1) \\ f(n+1) \cdot f(n) \end{pmatrix}$$

   Wir kodieren die Tupel mit der bijektiven Paarungsfunktion $c : \mathbb{N}^2 \to \mathbb{N}$ der Vorlesung und benützen die Projektionen p_1 und p_2 der Umkehrfunktion von c. Man beachte, dass sowohl c als auch die Projektionen p_1 und p_2 primitiv rekursiv sind.

   Wir betrachten die Funktion $g : \mathbb{N} \to \mathbb{N}$ mit

$$g(n) = c(f(n), f(n+1))$$

   und zeigen deren primitive Rekursivität mit erweiterten Rekursionsschemata wie folgt.

$$
\begin{array}{rcl}
g(0) & = & c(1,2)\,, \\
g(n+1) & = & c(\,p_2(g(n))\,,\, p_2(g(n)) \cdot p_1(g(n))\,)\,.
\end{array}
$$

   Nun gilt aber auch

$$f(n) = p_1(g(n))\,.$$

   Damit ist die primitive Rekursivität von f gezeigt.

2. Wir zeigen zunächst $f(n) > 1$ für alle $n \geq 1$ mittels geeigneter Induktion.

   $n = 1, n = 2$: Klar wegen $f(1) = 2 > 1$ und $f(2) = 2 > 1$.

   $(n-1, n) \Rightarrow (n+1)$ für $n \geq 2$: $f(n+1) = f(n) \cdot f(n-1) > 1 \cdot 1 = 1$.

   Nun folgt sofort für alle $n \geq 2$:

$$f(n+1) = f(n) \cdot f(n-1) > f(n)\,.$$

3. Ja, der Wertebereich ist entscheidbar, denn f ist streng monoton ($f(n) < f(n+1)$) für alle $n \geq 2$.

   Um zu prüfen, ob es zu $k \in \mathbb{N}$ ein $n \in \mathbb{N}$ gibt, so dass $f(n) = k$ gilt, bestimmt man $W_k = \{f(m) \mid m \leq f(k)\}$. Dann gilt $k \in W$ genau dann, wenn $k \in W_k$.