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

Sommersemester 2014
Übungsblatt 10
23. Juni 2014

Theoretische Informatik

Abgabetermin: 2. Juli 2014, 10 Uhr in die **THEO Briefkästen**

Hausaufgabe 1 (5 Punkte)

1. Geben Sie eine deterministische Turingmaschine $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ an, die für ein Eingabewort $w \in \{0,1\}^+$ eine Berechnung durchführt, so dass am Ende der Berechnung der Kopf am Anfang des Wortes w^R auf dem sonst leeren Band steht. (Erinnerung: w^R ist das gespiegelte Wort zu w.)

   Beschreiben Sie zunächst informell Ihre Lösungsidee!

2. Zeigen Sie durch Angabe einer geeigneten Konfigurationsfolge, dass Ihre Turingmaschine T die Eingabe 10 korrekt verarbeitet.

Hausaufgabe 2 (5 Punkte)

Geben Sie für die Sprache

$$L = \{ww\,;\, w \in \{0,1\}^*\}$$

einen linear beschränkten Automaten (LBA) M an, der L akzeptiert.

Hausaufgabe 3 (5 Punkte)

Seien K_1 eine kontextfreie Sprache und K_2, K_3 deterministische kontextfreie Sprachen über Σ. Zeigen Sie:

1. Für $\overline{K_2 \cap K_3}$ ist das Leerheitsproblem entscheidbar.
2. Für $\overline{K_1 \cap K_2}$ ist das Wortproblem entscheidbar.

Hausaufgabe 4 (5 Punkte)

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).

Zusatzaufgabe 7 (Wird nicht korrigiert)

Wir betrachten den Beweis zu dem Satz der Vorlesung, in dem eine k-Band-Turingmaschine M durch eine normale Turingmaschine M' simuliert wird. Geben Sie eine Abschätzung für die Anzahl der Zustände an, die M' haben muss.

Hinweis: *Die Vorbereitungsaufgaben bereiten die Tutoraufgaben vor und werden in der Zentralübung unterstützt. Tutoraufgaben werden in den Übungsgruppen bearbeitet. Hausaufgaben sollen selbstständig bearbeitet und zur Korrektur und Bewertung abgegeben werden.*

Vorbereitung 1

Vergleichen Sie die beiden folgenden `WHILE`-Programme:
`WHILE` $b_0 \wedge b_1$ `DO` P `END`
und
`WHILE` b_0 `DO` P `END; WHILE` $b_0 \wedge b_1$ `DO` P `END`.
Zeigen diese Programme das gleiche Verhalten? Begründung!

Vorbereitung 2

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

$$iszero(x) = \begin{cases} 1 & \text{falls } x = 0 \\ 0 & \text{sonst} \end{cases}, \qquad eq(x,y) = \begin{cases} 1 & \text{falls } x = y \\ 0 & \text{sonst} \end{cases}.$$

Vorbereitung 3

Sei $f(x,y)$ primitiv rekursiv. Zeigen Sie mit Hilfe der Projektionsfunktionen π_i^k zusammen mit der (nicht erweiterten) Komposition, dass die Funktion g mit $g(x,y) = f(y,x)$ für alle $x, y \in \mathbb{N}$ ebenfalls primitiv rekursiv ist.

Vorbereitung 4

Wir bezeichnen f als eine erweiterte Komposition der Funktionen $g_1, \ldots, g_k$, falls $f(x_1, \ldots, x_n) = t$, so dass t ein Ausdruck ist, der nur aus den Funktionen $g_1, \ldots, g_k$ und den Variablen $x_1, \ldots, x_n$ besteht.

Sei t_0 ein funktionaler Ausdruck, der nur primitiv-rekursive Funktionen und Variable x_i enthält. t enthalte nur $f(m, \overline{x})$ mit einem Variablenvektor $\overline{x}$, primitiv-rekursive Funktionen, m und Variable x_i. Dann heißen die Gleichungen $f(0, \overline{x}) = t_0$, $f(m+1, \overline{x}) = t$ das erweiterte Schema der primitiven Rekursion.

Man zeige:

1. Eine erweiterte Komposition von primitiv-rekursiven Funktionen ist wieder primitiv-rekursiv.
2. Das erweiterte Schema der primitiven Rekursion führt nicht aus der Menge der primitiv-rekursiven Rekursionen heraus.

Vorbereitung 5

Zeigen Sie, dass man die folgende Anweisung durch ein LOOP-Programm simulieren kann, das kein IF-Konstrukt enthält: **IF** $x_i \leq x_j$ **THEN** P_1 **ELSE** P_2 **END**.

Tutoraufgabe 1

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

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

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

Tutoraufgabe 2

Der Binomialkoeffizient $binom(n, m) = \binom{n}{m}$ ist eine Funktion von $\mathbb{N}^2$ in $\mathbb{N}$ mit den Eigenschaften $\binom{n}{0} = 1$, $\binom{0}{m} = 0$ für $n, m \in \mathbb{N}$ mit $m > 0$. Der Binomialkoeffizient erfüllt für alle $n, m \in \mathbb{N}$ die Rekursionsgleichung

$$\binom{n+1}{m+1} = \binom{n}{m+1} + \binom{n}{m}\,.$$

Für jede natürliche Zahl m_0 betrachten wir die Funktion $b_{m_0} : \mathbb{N} \to \mathbb{N}$ mit $b_{m_0}(n) = binom(n, m_0)$. Zeigen Sie durch Induktion über m_0, dass alle Funktionen $b_{m_0}(n)$ mit $m_0 \in \mathbb{N}$ primitiv rekursiv (als Funktion in n) sind.

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, indem Sie ein LOOP-Programm angeben.
2. Zeigen Sie durch Rückführung auf die Definition, dass f primitiv-rekursiv ist.

Tutoraufgabe 4

Wir wollen untersuchen, ob sich (ähnlich wie bei den WHILE-Programmen) auch jedes LOOP-Programm in eine „Normalform“

`LOOP` X `DO` P `END`

bringen lässt, so dass P keine Schleifen mehr enthält.
Wir bezeichnen mit $V(P)$ die Menge der Variablen, die in P vorkommen (diese Menge ist stets endlich). Für einen gegebenen Programmlauf bezeichnen wir mit $[x]$ den Wert der Variablen x beim Start des Programms, und mit $[x]'$ den Wert der Variablen nach Programmende.

1. Zeigen Sie: Für jedes LOOP-Programm P ohne Schleifen gibt es eine Konstante k, so dass gilt:

$$\max_{x \in V(P)} [x]' \leq \max_{x \in V(P)} [x] + k\,.$$

2. Zeigen Sie, dass es kein LOOP-Programm in Normalform geben kann, welches die Quadratfunktion $n \mapsto n^2$ berechnet.