Kurs 1653
Einführung in die Theoretische Informatik A
Lösungshinweise zur Klausur vom 21. Februar 2004

Aufgabe 1

(a) Sei $n \in \mathbb{N}$. Wir beweisen die Aussage durch vollständige Induktion über $t \in \mathbb{N}$.
$\underline{t = 0}$ Für $t = 0$ gilt

$$\mathrm{ES}^{3t}(4, (1, n, 2, 0, 0, 0, ...)) = (4, (1, n, 2, 0, 0, 0, ...))$$

und damit die gewünschte Behauptung.
$\underline{t \to t+1}$ Sei $2^{t+1} \leq n$. Dann ist insbesondere $2^t \leq n$ und mit der Induktionsvoraussetzung erhalten wir

$$\begin{aligned}
& \mathrm{ES}^{3(t+1)}(4, (1, n, 2, 0, 0, 0, ...)) \\
= \; & \mathrm{ES}^{3}(4, (t+1, n, 2^{t+1}, 0, 0, 0, ...)) \\
= \; & \mathrm{ES}^{2}(2, (t+1, n, 2^{t+1}, 0, 0, 0, ...)) \text{ da } 2^{t+1} \leq n \\
= \; & \mathrm{ES}(3, (t+2, n, 2^{t+1}, 0, 0, 0, ...)) \\
= \; & (4, (t+2, n, 2^{t+2}, 0, 0, 0, ...)),
\end{aligned}$$

also die Behauptung für $t + 1$. Damit ist die Induktion abgeschlossen.

(b) Sei $n \in \mathbb{N}$. Wir erhalten

$$\begin{aligned}
& \mathrm{ES}^{3\log(n)+1}(1, (0, n, 0, 0, 0, ...)) \\
= \; & \mathrm{ES}^{3(\log(n)-1)+3}(2, (0, n, 1, 0, 0, ...)) \\
= \; & \mathrm{ES}^{3(\log(n)-1)+2}(3, (1, n, 1, 0, 0, ...)) \\
= \; & \mathrm{ES}^{3(\log(n)-1)+1}(4, (1, n, 2, 0, 0, ...)) \\
= \; & \mathrm{ES}(4, (\log(n), n, 2^{\log(n)}, 0, 0, ...)) \\
& \text{dies gilt für } n \in \{0, 1\} \text{ da dann } \log(n) = 1 \text{ ist und} \\
& \text{für } n > 1 \text{ nach (a), da dann } 2^{\log(n)-1} \leq n \text{ nach Definition von log} \\
= \; & (5, (\log(n), n, 2^{\log(n)}, 0, 0, ...)) \\
& \text{da } 2^{\log(n)} > n \text{ nach Definition von log.}
\end{aligned}$$

Insgesamt folgt daraus $f_M(n) = \log(n)$ für alle $n \in \mathbb{N}$, also $f_M = \log$.

(c) Da alle im Flußdiagramm von M vorkommenden Tests und Funktionen berechenbar sind (insbesondere auch diejenigen an den Marken 3 und 4), folgt mit dem Satz über die verallgemeinerten Registermaschinen, daß die Funktion f_M berechenbar ist. Daher ist nach (b) die Funktion $\log = f_M$ berechenbar.

Aufgabe 2

(a) Mit dem Gleichheitstest $g : \mathbb{N}^2 \to \mathbb{N}$ und der Exponentialfunktion $e : \mathbb{N} \to \mathbb{N}$ zur Basis 2 aus dem Anhang der Klausur erhalten wir

$$t(n) = \sum_{i=0}^{n} g(n, 2^i) = \sum_{i=0}^{n} g(n, e(i))$$

für alle $n \in \mathbb{N}$. Man überlegt sich, daß diese Identität gilt, da n Zweierpotenz ist, genau dann wenn für ein $i = 0, ..., n$ gilt, daß $n = 2^i$ ist. Ein formaler Beweis läßt sich zum Beispiel durch vollständige Induktion führen. Wir nutzen nun obige Gleichung aus, um zu zeigen, daß t primitiv-rekursiv ist. Dazu definieren wir eine Funktion $f : \mathbb{N}^2 \to \mathbb{N}$ durch

$$f(n, k) := \sum_{i=0}^{k} g(n, e(i))$$

für alle $n, k \in \mathbb{N}$. Dies führt zu der folgenden Rekursionsgleichung

$$\begin{cases} f(n, 0) & = & g(n, 1) \\ f(n, k+1) & = & g(n, e(k+1)) + f(n, k) \end{cases}$$

Um dies in der gewünschten Form auszudrücken, definieren wir eine Funktion $g' : \mathbb{N} \to \mathbb{N}$ durch

$$g'(n) := g(n, 1) = \mathrm{Sub}(g, \mathrm{pr}_1^{(1)}, c_1^{(1)})(n)$$

für alle $n \in \mathbb{N}$ (mit den Bezeichnungen aus dem Anhang der Klausur). Weiterhin definieren wir $h : \mathbb{N}^3 \to \mathbb{N}$ durch

$$\begin{aligned} h(n, k, i) &:= g(n, e(k+1)) + i = s(g(n, eS(k)), i) \\ &= s(g(\mathrm{pr}_1^{(3)}(n, k, i), eS\mathrm{pr}_2^{(3)}(n, k, i)), \mathrm{pr}_3^{(3)}(n, k, i)) \\ &= \mathrm{Sub}(s, \mathrm{Sub}(g, \mathrm{pr}_1^{(3)}, \mathrm{Sub}(e, \mathrm{Sub}(S, \mathrm{pr}_2^{(3)}))), \mathrm{pr}_3^{(3)})(n, k, i) \end{aligned}$$

für alle $n, k, i \in \mathbb{N}$. Dies zeigt, daß g' und h primitiv-rekursiv sind. Damit ist auch $f = \mathrm{Prk}(g', h)$ eine primitiv-rekursive Funktion. Wegen $t(n) = f(n, n) = \mathrm{Sub}(f, \mathrm{pr}_1^{(1)}, \mathrm{pr}_1^{(1)})(n)$ ist damit auch t primitiv-rekursiv.

(b) Wir zeigen nun mit Hilfe der primitiv-rekursiven Funktion t aus (a), daß auch L primitiv-rekursiv ist. Dazu nutzen wir die Rekursionsgleichung

$$\begin{cases} L(0) & = & 1 \\ L(n+1) & = & L(n) + t(n+1) \end{cases}$$

aus, die für alle $n \in \mathbb{N}$ erfüllt ist (wie man zum Beispiel durch vollständige Induktion zeigen kann). Um die gewünschte Form für das Schema

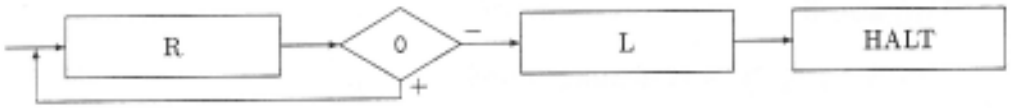
der primitiven Rekursion zu erhalten, schreiben wir $L(0) = 1 = c_1^{(0)}()$ und wir definieren $h' : \mathbb{N}^2 \to \mathbb{N}$ durch

$$\begin{aligned} h'(n,i) &:= i + t(n+1) = s(i, tS(n)) \\ &= s(\mathrm{pr}_2^{(2)}(n,i), tS\mathrm{pr}_1^{(2)}(n,i)) \\ &= \mathrm{Sub}(s, \mathrm{pr}_2^{(2)}, \mathrm{Sub}(t, \mathrm{Sub}(S, \mathrm{pr}_1^{(2)})))(n,i) \end{aligned}$$

für alle $n, i \in \mathbb{N}$. Dann ist h' primitiv-rekursiv (da t nach (a) primitiv-rekursiv ist) und somit $L = \mathrm{Prk}(c_1^{(0)}, h')$ auch primitiv-rekursiv.

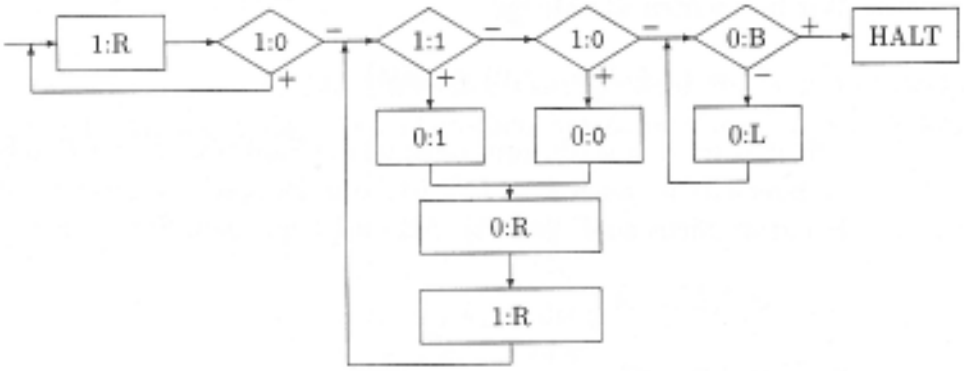
Aufgabe 3

(a) Wir betrachten die einstellige Bandmaschine M mit dem Ein- und Ausgabealphabet Σ, die durch folgendes Flußdiagramm gegeben ist:



Offenbar überspringt die Maschine alle führenden Nullen in der Eingabe und gibt das restliche Wort als Ausgabe aus. Damit berechnet die Bandmaschine M genau die gewünschte Funktion f.

(b) Wir betrachten die einstellige Turingmaschine M' mit dem Ein- und Ausgabealphabet Σ, die durch folgendes Flußdiagramm gegeben ist:



Diese Maschine überspringt zunächst alle führenden Nullen der Eingabe, kopiert dann alle folgenden Einsen und Nullen der Reihe nach auf das Ausgabeband und geht schließlich an den Anfang des Ausgabebandes zurück. Damit berechnet auch diese Turingmaschine M' genau die gewünschte Funktion f.

Aufgabe 4

Sei $f :\subseteq \mathbb{N}^2 \to \mathbb{N}$ definiert durch

$$f(\langle i, j\rangle, n) := u_\varphi(i, \langle j, n\rangle)$$

für alle $i, j, n \in \mathbb{N}$. Gemäß utm-Theorem ist f dann berechenbar und gemäß smn-Theorem gibt es ein $s \in R^{(1)}$, so daß

$$\varphi_{s(k)}(n) = f(k, n)$$

für alle $n, k \in \mathbb{N}$ gilt. Wir erhalten

$$\varphi_{s(i,j)}(n) = f(\langle i, j\rangle, n) = u_\varphi(i, \langle j, n\rangle) = \varphi_i\langle j, n\rangle$$

für alle $i, j, n \in \mathbb{N}$.

Aufgabe 5

(a) Sei $f :\subseteq \mathbb{N} \to \mathbb{N}$ definiert durch

$$f(i) := \begin{cases} 0 & \text{falls } u_\varphi(i, 0) \leq 2004 \\ \text{div} & \text{sonst} \end{cases}$$

für alle $i \in \mathbb{N}$. Gemäß utm-Theorem ist f eine berechenbare Funktion, wie man zum Beispiel mit Hilfe einer verallgemeinerten Registermaschine zeigen kann. Es gilt $\text{Def}(f) = A$. Also ist A rekursiv-aufzählbar.

(b) Wir betrachten die Menge

$$F := \{f \in P^{(1)} \mid f(0) \leq 2004\}.$$

Offenbar ist $F \neq \emptyset$ (da zum Beispiel die Funktion $c_0^{(1)}$ zu F gehört) und andererseits ist auch $F \neq P^{(1)}$ (da zum Beispiel die nirgends definierte Funktion nicht zu F gehört). Also folgt mit dem Satz von Rice, daß

$$\varphi^{-1}[F] = A$$

nicht rekursiv ist.

Aufgabe 6

(a) Wir definieren eine Funktion $f :\subseteq \mathbb{N} \to \mathbb{N}$ durch

$$f\langle i, n\rangle := u_\varphi(i, n)$$

für alle $i, n \in \mathbb{N}$. Nach dem utm-Theorem ist f berechenbar und es gilt $\mathrm{Def}(f) = H$. Damit ist H rekursiv-aufzählbar.

(b) Nach (a) ist H rekursiv-aufzählbar. Gilt nun $A \leq H$, so folgt mit Satz 9.3.5.4 (Satz 9.16(4) in älteren Kursversionen), daß auch A rekursiv-aufzählbar ist.

(c) Sei nun A rekursiv-aufzählbar. Dann gibt es ein $i \in \mathbb{N}$ mit $\mathrm{Def}(\varphi_i) = A$. Die Funktion $f : \mathbb{N} \to \mathbb{N}$ mit $f(n) = \langle i, n\rangle$ für alle $n \in \mathbb{N}$ ist offenbar berechenbar und es gilt

$$\begin{aligned} f(n) \in H &\iff \langle i, n\rangle \in H \\ &\iff n \in \mathrm{Def}(\varphi_i) = A \end{aligned}$$

für alle $n \in \mathbb{N}$, d.h. $A \leq H$.

Aufgabe 7

Wir definieren eine Funktion $F : \mathbb{N}^2 \to \mathbb{N}$ durch

$$F(\langle i, j, k\rangle, \langle a, b, c\rangle) := \begin{cases} \langle i, j, k\rangle & \text{falls } ic + i + bk + b \leq jc + j + ak + a \\ \langle a, b, c\rangle & \text{sonst} \end{cases}$$

für alle $i, j, k, a, b, c \in \mathbb{N}$. Mit Hilfe einer verallgemeinerten Registermaschine läßt sich leicht zeigen, daß F eine berechenbare Zahlenfunktion ist. Weiterhin gilt

$$\begin{aligned} \nu_{\mathbb{Q}} F(\langle i, j, k\rangle, \langle a, b, c\rangle) &= \begin{cases} \nu_{\mathbb{Q}}\langle i, j, k\rangle & \text{falls } ic + i + bk + b \leq jc + j + ak + a \\ \nu_{\mathbb{Q}}\langle a, b, c\rangle & \text{sonst} \end{cases} \\ &= \begin{cases} \nu_{\mathbb{Q}}\langle i, j, k\rangle & \text{falls } (i - j)(c + 1) \leq (a - b)(k + 1) \\ \nu_{\mathbb{Q}}\langle a, b, c\rangle & \text{sonst} \end{cases} \\ &= \begin{cases} \nu_{\mathbb{Q}}\langle i, j, k\rangle & \text{falls } \nu_{\mathbb{Q}}\langle i, j, k\rangle \leq \nu_{\mathbb{Q}}\langle a, b, c\rangle \\ \nu_{\mathbb{Q}}\langle a, b, c\rangle & \text{sonst} \end{cases} \\ &= \min\{\nu_{\mathbb{Q}}\langle i, j, k\rangle, \nu_{\mathbb{Q}}\langle a, b, c\rangle\} \\ &= f(\nu_{\mathbb{Q}}\langle i, j, k\rangle, \nu_{\mathbb{Q}}\langle a, b, c\rangle) \end{aligned}$$

für alle $i, j, k, a, b, c \in \mathbb{N}$, d.h. f ist $(\nu_{\mathbb{Q}}, \nu_{\mathbb{Q}}, \nu_{\mathbb{Q}})$–berechenbar.

Aufgabe 8

Seien $w_0, w_1 \in \mathrm{Def}(\nu_{\mathrm{rat}})$ Wörter mit $\nu_{\mathrm{rat}}(w_0) = 0$ und $\nu_{\mathrm{rat}}(w_1) = 1$ (dies gilt beispielsweise für $w_0 := 0¢1$ und $w_1 := 1¢1$). Wir konstruieren eine Typ-2-Maschine M, die $t :\subseteq \mathbb{R} \to \mathbb{R}$ berechnet. Die Maschine M soll sich bei Eingabe einer Folge $p = u_0\#u_1\#u_2\#... \in \mathrm{Def}(\rho)$ mit $x := \rho(p)$ wie folgt verhalten: die Wörter $u_0, u_1, ...$ auf dem Eingabeband werden eingelesen, solange bis ein $i \in \mathbb{N}$ gefunden ist, für das gilt

$$\nu_{\mathrm{rat}}(u_i) < -2^{-i} \text{ oder } \nu_{\mathrm{rat}}(u_i) > 2^{-i}.$$

Trifft der erste Fall zuerst zu, dann schreibt die Maschine $q = w_0\#w_0\#w_0\#...$ auf das Ausgabeband (für dieses q gilt offenbar $\rho(q) = 0$); trifft der zweite Fall zuerst zu, dann schreibt die Maschine $q = w_1\#w_1\#w_1\#...$ auf das Ausgabeband (und hier gilt analog $\rho(q) = 1$). Wird kein passendes $i \in \mathbb{N}$ gefunden, dann sucht die Maschine unendlich lange und schreibt gar nichts auf das Ausgabeband. Wir müssen noch begründen, daß die Maschine M korrekt arbeitet, daß also

$$t\rho(p) = \rho(q)$$

für jede Eingabe $p \in \mathrm{Def}(t\rho)$ und die zugehörige Ausgabe $q = f_M(p)$ gilt. Zunächst erwähnen wir, daß gemäß der Definition der Cauchy-Darstellung ρ der reellen Zahlen für jedes $p = u_0\#u_1\#u_2\#... \in \mathrm{Def}(\rho)$ und für alle $i \in \mathbb{N}$ gilt $|x - \nu_{\mathrm{rat}}(u_i)| \leq 2^{-i}$. Weiterhin gilt

$$|x - \nu_{\mathrm{rat}}(u_i)| \leq 2^{-i} \text{ und } \nu_{\mathrm{rat}}(u_i) < -2^{-i} \Longrightarrow x < 0$$

und analog

$$|x - \nu_{\mathrm{rat}}(u_i)| \leq 2^{-i} \text{ und } \nu_{\mathrm{rat}}(u_i) > 2^{-i} \Longrightarrow x > 0.$$

Daraus folgt bereits, daß wenn die Maschine M zu $p \in \mathrm{Def}(t\rho)$ ein passendes $i \in \mathbb{N}$ findet und folglich q gemäß dem obigen Algorithmus ausgibt, dann gilt auch

$$t\rho(p) = \begin{cases} 0 & \text{falls } \rho(p) = x < 0 \\ 1 & \text{falls } \rho(p) = x > 0 \end{cases} = \rho(q).$$

Zu zeigen bleibt, daß die Maschine M für alle $p = u_0\#u_1\#u_2\#... \in \mathrm{Def}(t\rho)$ wirklich ein passendes $i \in \mathbb{N}$ findet. Ist aber $\rho(p) = x < 0$, dann gibt es auch ein $i \in \mathbb{N}$, so daß $x < -2^{-i+1}$ gilt. Daraus folgt dann

$$\nu_{\mathrm{rat}}(u_i) \leq x + 2^{-i} < -2^{-i}.$$

Dieses i wäre also passend und würde folglich von der Maschine M gefunden. Im Falle $\rho(p) = x > 0$ zeigt man analog, daß dann ein $i \in \mathbb{N}$ mit $\nu_{\mathrm{rat}}(u_i) > 2^{-i}$ existiert. Insgesamt folgt daraus, daß M die gewünschte Funktion t berechnet.