Universität Karlsruhe (TH)
Lehrstuhl Informatik für Ingenieure und Naturwissenschaftler
Dr. Thomas Worsch
Matthias Schulz

30. Juni 2008

Einführung in die Informatik

Lösungen zu Übungsblatt 10

– Registermaschinen –

Aufgabe 1:

a) $(s_2)_2$ setzt Register 2 auf 0. Die Schleife $(a_2(s_1)_1)_1$ wird genau einmal durchlaufen, falls x "wahr" ist, sonst nicht. Nach dieser Schleife ist Register 1 mit 0 belegt und Register 2 mit 1, falls x "wahr" ist und 0 sonst. $(s_2a_1)_2$ verschiebt den Inhalt von Register 2 nach Register 1.

Das Programm gibt 1 aus, falls x "wahr" ist, 0 sonst.

(Anders ausgedrückt: Das Programm berechnet $Sg(x)$).

b) Wir stellen eine Tabelle auf:

Programm	R_1	R_2	R_3	R_4	R_5
	x	y	?	?	?
$(s_3)_3(a_3(s_1)_1)_1(s_3a_1)_3$	$Sg(x)$	y	0	?	?
$(s_4)_4(a_4(s_2)_2)_2(s_4a_2)_4$	$Sg(x)$	$Sg(y)$	0	0	?
$(s_5)_5(s_1a_4a_5)_1$	0	$Sg(y)$	0	$Sg(x)$	$Sg(x)$
$(s_2a_3s_4)_2$	0	0	$Sg(y)$	$Sg(x) \dot{-} Sg(y)$	$Sg(x)$
$(s_3s_5)_5$	0	0	$Sg(y) \dot{-} Sg(x)$	$Sg(x) \dot{-} Sg(y)$	0
$a_5(s_5s_4)_4(s_5s_3)_3$	0	0	0	0	$f(x,y)$

Dabei ist $f(x,y) = 1 \dot{-} (Sg(x) \dot{-} Sg(y)) \dot{-} (Sg(y) \dot{-} Sg(x))$.

Dieser Ausdruck ist genau dann 1, wenn $Sg(x) = Sg(y)$ gilt und 0 sonst.

Dies entspricht der Äquivalenz: $f(x,y)$ ist "wahr" genau dann, wenn entweder sowohl x als auch y "wahr" sind oder sowohl x als auch y "falsch" sind.

c) Zuerst wird der Inhalt von Register 1 nach Register 2 kopiert: $(s_2)_2(s_3)_3(s_1a_2a_3)_1(s_3a_1)_3$. Danach wird der Inhalt von Register 2 um 1 verringert, **falls möglich**, und danach um 1 erhöht: s_2a_2

Falls die Eingabe x 0 war, steht nun im Register 2 eine 1, ansonsten wurde Register 2 durch die beiden Operationen unverändert gelassen.

Nun wird Register 1 von Register 2 abgezogen, und in Register 2 steht eine 0, falls $x > 0$ gilt, und eine 1 sonst: $(s_1 s_2 a_3)_1 (s_3 a_1)_3$

Dies entspricht bei unserer Interpretation als "wahr" oder "falsch" gerade der Negation von x.

Ein kürzeres Programm: Wir berechnen $1 \dot{-} x$ in Register 2.:

$(s_2)_2 (s_3)_3 a_2 (s_1 s_2 a_3)_1 (s_3 a_1)_3$.

Aufgabe 2:

a) Idee: Wir setzen Register 2 auf $\mathsf{fib}(0)$ und Register 3 auf $\mathsf{fib}(1)$. Solange der Inhalt von Register 1 nicht 0 ist, wird Register 1 um 1 verringert, Register 2 auf den Inhalt von Register 3 gesetzt und Register 3 auf die Summe der (ursprünglichen) Register 2 und 3 gesetzt.

Wenn Register 1 0 erreicht hat, steht dann im zweiten Register nach Definition von fib die Zahl $\mathsf{fib}(n)$.

Nun wird Register 3 auf 0 gesetzt, und eine anfangs angefertigte "Sicherheitskopie" von x wird in Register 1 geschrieben.

Code (die Eingabe heiße n):

$(s_1 a_4 a_5)_1 (s_5 a_1)_5$: Kopieren n nach Register 4
$a_3 (s_1 (s_2 a_5)_2 (s_3 a_2 a_5)_3 (s_5 a_3)_5)_1 (s_3)_3$: Berechnen von $\mathsf{fib}(n)$ und Löschen von Register 3
$(s_4 a_1)_4$: Verschieben von n nach Register 1.

b) Idee: Wir kopieren n in Register 3 und setzen Register 2 auf 1. Solange Register 4 keine 0 enthält, wird in Register 2 das Produkt der Inhalte von Register 2 und Register 3 in Register 4 berechnet und von dort nach Register 2 kopiert und danach Register 3 um 1 verringert.

Code:

$(s_1 a_3 a_4)_1 (s_4 a_1)_4$: Kopieren von n nach Register 3
a_2: Register 2 auf 1 setzen
$(((s_3 a_4 a_5)_3 (s_5 a_3)_5 s_2)_2 (s_4 a_2)_4 s_3)_3$: Register 2 mit Register 3 multiplizieren, danach Register 3 um 1 verringern, bis Register 3 0 enthält.

c) Idee: Wir kopieren n in Register 3. Nun addieren wir Register 1 zu Register 2 und verringern Register 3 um 1, bis Register 3 0 erreicht. Dann steht in Register 2 der Wert n^2.

Code:

$(s_1 a_2 a_3)_1 (s_2 a_1)_2$: n kopieren
$((s_1 a_2 a_4)_1 (s_4 a_1)_4 s_3)_3$: n mal n zu 0 addieren.

d) Idee: Wir berechnen nacheinander für die Werte 0, 1, ...in Register 2 das jeweilige Quadrat in Register 3. In Register 4 berechnen wir die Differenz zwischen Register 1 und Register 3 und in Register 5 die Differenz zwischen Register 3 und Register 1.

Falls Register 4 und Register 5 beide 0 sind, bricht die Schleife ab und das Register 3 wird auf 0 gesetzt. Falls Register 4 eine 0 enthält und Register 5 nicht, wird Register 2 um 1 verringert und die Register 3 und 5 auf 0 gesetzt.

Code:

a_4: Damit Schleife betreten wird
$((s_2a_3a_4)_2(s_3a_2)_3((s_2a_3a_5)_2(s_5a_2)_5s_4)_4$: Register 2 quadrieren
$(s_1a_4a_6)_1(s_3a_5s_4)_3(s_6s_5a_1)_6$: Register 1-Register 3 und Register 3-Register 1 berechnen
$a_2)_4$: Wiederholen, bis Register 4 0 enthält
$s_2(s_2(s_5)_5)_5(s_3)_3$: Register 2 entsprechend Register 5 verringern, Register 3 auf 0 setzen.

e) Code:

$a_3(s_2a_4a_5)_2(s_5a_2)_2$: Register 2 auf 1 setzen, Register 3 auf m
$((s_2(s_1a_4a_5)_1(s_5a_1)_5)_2$: Multipliziere Register 2 mit n
$(s_4a_2)_4$: Verschiebe Produkt nach Register 2
$s_3)_3$: Wiederhole Multiplikation m mal.

f) Code:

$(s_1a_4a_5)_1(s_5a_1)_5$: Kopieren von n
$(s_2a_5a_6)_2(s_6a_2)_6$: Kopieren von m
$(s_4((s_5a_6)_5a_3)_5(s_6a_5)_5s_5)_4$: Register 3 wird erhöht, falls Register 4 und 5 jeweils Werte größer 0 enthalten. Register 4 und Register 5 werden pro Schleifendurchlauf um 1 verringert. $(s_5)_5$: Register 5 wird auf 0 gesetzt.

g) Code:

$(s_1a_4a_5)_1(s_5a_1)_5$: Kopieren von n
$(s_2a_5a_6)_2(s_6a_2)_6$: Kopieren von m
$(s_4s_5a_3)_4(s_5a_3)_5$

Aufgabe 3:

a) Sei P ein Registermaschinenprogramm. Für $y \in \mathbb{N}$ definieren wir P_y und P'_y als $P_y = (s_1)_1[a_1]^yP$ (das heißt, wir löschen die Eingabe, setzen sie danach auf y und lassen danach P laufen) und $P' = a_1(a_1)_1$.

Falls P bei Eingabe von y nicht hält, sind P_y und P' äquivalent, da in diesem Fall P_y für keine Eingabe hält, ebenso wie P'.

Falls P bei Eingabe von y nicht hält, sind P_y und P' nicht äquivalent.

Ein Programm, das die Äquivalenz zweier beliebiger Programme entscheiden könnte, könnte somit auch das (unentscheidbare) Halteproblem lösen, indem man zunächst den Code von P_y berechnet (*das geht nach Church-Turing-These!*) und dann P_y mit P' vergleicht. Aus diesem Grund kann es so ein Registermaschinenprogramm nicht geben.

b) Möglichkeit 1: Wir ordnen den Symbolen $a, s, (,)$ die Zahlen 10, 11, 12 und 13 zu und fassen ein Programm als Zahl im 14er-System auf.

So hätte zum Beispiel das Programm $(a_1)_1$ den "Wert" $1 + 13 \cdot 14 + 1 \cdot 196 + 10 \cdot 14^3 + 12 \cdot 14^4 = 488811$.

Möglichkeit 2: Wir nummerieren die einzelnen Operationen $a_i, s_i, (,)_i$ des Programms durch. Die Schleife interpretieren wir als Sprung vom Befehl (zum entsprechenden Befehl $)_i$, wo Register i auf 0 überprüft wird und dann entweder die nächste Operation ausgeführt oder zum Befehl nach der geöffneten Klammer zurück gesprungen wird.

Jede der vier Operationen besitzt höchstens zwei Argumente (Addieren und Subtrahieren: Das jeweilige Register; unbedingter Sprung: Sprungziel; bedingter Sprung: Register, das auf 0 überprüft wird, und Sprungziel).

Wir merken uns für jede Operation die Tripel (OpCode, Arg1, Arg2), wobei OpCode die Nummer des Befehls (1 für a, 2 für s und so weiter) und Arg1 und Arg2 die Argumente sind. Wir erstellen eine (beliebig lange) Liste (n_i) paarweise teilerfremde Zahlen und bilden die Befehlsfolge (OpCode_1, Arg1_1, Arg2_1)(OpCode_2, Arg1_2, Arg2_2), ...auf die Zahl $n_1^{\text{OpCode}_1} n_2^{\text{Arg1}_1} \cdots$ ab. (So eine Codierung eignet sich besonders gut zur Simulation einer Registermaschine.)

Jede Codierung sollte injektiv sein (was obige Codierungen auch sind), damit sich aus dem Code das zugehörige Programm eindeutig herauslesen lässt. Umkehrbar ist keine der beiden Codierungen, da nicht alle Zahlen zu wohlgeformten Programmen führen:

Möglichkeit 1: Die Zahl 12 würde das Nicht-Programm (repräsentieren.

Möglichkeit 2: Die Zahl n_1^5 beschreibt kein Programm, da der OpCode keinen Befehl beschreibt.