Prof. Dr. Manfred Schmidt-Schauß
Künstliche Intelligenz/Softwaretechnologie

Fachbereich Informatik (20)
Johann Wolfgang Goethe-Universität Frankfurt am Main

Praktische Informatik 1:

Wintersemester 1997/98

Übungsblatt Nr. 3

Aufgabe 1 (10 Punkte)

Sei $A(x, y)$ die Ackermann-Funktion aus der Vorlesung. Zeigen Sie, daß

$$
\begin{aligned}
A(x,1) &= 2x \\
A(x,2) &= 2^x \\
A(x,3) &= 2^{2^{\cdot^{\cdot^{2}}}} \quad \text{(ein “Stapel” von } x \text{ zweien)}
\end{aligned}
$$

Aufgabe 2 (10 Punkte)

Füllen Sie die folgende Wertetabelle aus.

$x \backslash y$	1	2	3	4	5
1					
2					
3					
4				•	•

Schalten Sie die Statistik-Ausgabe im `Hugs` ein (`:s +s`) und beobachten Sie wie sich die Zahl der Reduktionen bei $A(x, 2)$ mit wachsendem x verändert. Wieviele Reduktionen schätzen Sie für $A(14, 2)$? Wieviele Reduktionen schätzen Sie für $A(4, 3)$?

Aufgabe 3 (10 Punkte)

$g \in O(f)$	gdw. es gibt $r > 0$ und für alle bis auf endlich viele n gilt $g(n) < r * f(n)$
$g \in o(f)$	gdw. für alle $r > 0$ und für alle bis auf endlich viele n gilt $g(n) < r * f(n)$
$g \in \Omega_\infty(f)$	gdw. es gibt $r > 0$ und für unendlich viele n gilt $g(n) > r * f(n)$
$g \in \omega(f)$	gdw. für alle $r > 0$ und für unendlich viele n gilt $g(n) > r * f(n)$
$g \in \Theta(f)$	gdw. es gibt $r_1, r_2 > 0$ und für alle bis auf endlich viele n gilt $r_1 * f(n) \leq g(n) \leq r_2 * f(n)$

Zeigen Sie

1. $g \in o(f) \iff \lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$
2. $g \in \Theta(f) \iff g \in O(f)$ und $f \in O(g)$

3. $g \in O(\log_2 n) \iff$ für alle $k \geq 2 : g \in O(\log_k n)$

4. Es ist $\log^* n = \min\{i | \underbrace{\log(\log(\ldots}_{i} n) \ldots) \leq 1\}$. Zeigen Sie $\frac{n}{\log^* n} \in o(n)$.

Aufgabe 4 (10 Punkte)

Angenommen Sie haben Programme, deren Resourcenbedarf in Abhängigkeit von der Eingabelänge n sich wie unten angegeben verhält. Wie groß kann die Eingabe sein, die Sie mit doppelter Rechnerkapazität bearbeiten können?

Resourcenbedarf	neue Eingabegröße
$\log_2 n$	
n	
n^k	
2^n	

Aufgabe 5 (10 Punkte)

Die Binomialkoeffizienten kann man auf viele verschiedene Weisen berechnen. Unter anderem als $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ und als $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$. Wobei $\binom{n}{0} = 1$, $\binom{n}{1} = n$ und $\binom{n}{n} = 1$. Programmieren Sie beide Berechnungen in `Hugs` und beobachten Sie, wie sich die Zahl der Reduktionen in Abhängigkeit von der Eingabe verhält. Erklären Sie Ihre Beobachtung.