RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
AACHEN

LEHR- UND FORSCHUNGSGEBIET INFORMATIK II
RWTH Aachen · D-52056 Aachen · GERMANY
http://www-i2.informatik.rwth-aachen.de/lufgi2

Prof. Dr. J. Giesl
D. Dlugosz

11. Übung zu „Automatisierte Programmverifikation“, SS 03
Abgabe: Mi, 23.07.03, in der Frontalübung

Aufgabe 1 (2 Punkte)

Sei P ein Programm und seien $s, t \in \mathcal{T}(\Sigma, \mathcal{V})$ und $s^*, t^* \in \mathcal{T}(\Sigma, \mathcal{V})^*$. Wir betrachten die Relation $\succ$, für die jeweils gilt

a) $f(s^*) \succ g(t^*)$ gdw. $f \neq g$ und g tritt im Algorithmus von f auf

b) $s \succ t$ gdw. $\mathcal{V}(t) \subsetneqq \mathcal{V}(s)$

Untersuchen Sie die Relation $\succ$ auf Fundiertheit, Transitivität, Monotonie und Stabilität.

Aufgabe 2 (2 Punkte)

Wir modifizieren die Definition der Einbettungsordnung $\succ_{emb}$ (vgl. Def. 6.1.5) wie folgt:

Für eine Signatur Σ ist die Relation $\succ$ über $\mathcal{T}(\Sigma, \mathcal{V})$ definiert als $s \succ t$ gdw.

- *t ist ein echter Teilterm von s oder*
- *$s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$, $s_i \succ t_i$ für ein $i \in \{1, \ldots, n\}$ und $s_j \succeq t_j$ für alle $j \in \{1, \ldots, n\}$ mit $j \neq i$.*

Hierbei bezeichnet $\succeq$ die Relation mit $s \succeq t$ gdw. $s \succ t$ oder $s = t$.

Beweisen oder widerlegen Sie, dass die Relation $\succ$ eine Reduktionsrelation oder eine Reduktionsordnung ist.

Vergleichen Sie die Relation $\succ$ mit der Einbettungsordnung $\succ_{emb}$. Welche der beiden Relationen ist für Terminierungsbeweise besser geeignet? Begründen Sie Ihre Antwort.

Aufgabe 3 (4 Punkte)

Uns stehen die Datenstruktur number und die folgenden Datenstrukturen und Algorithmen zur Verfügung:

structure sexpr

nil	:	sexpr
atom	:	number $\rightarrow$ sexpr
cons	:	sexpr $\times$ sexpr $\rightarrow$ sexpr

function subtract : number $\times$ number $\rightarrow$ number

subtract$(x, \mathcal{O})$	$\equiv$	x
subtract$(\mathcal{O}, \text{succ}(y))$	$\equiv$	$\mathcal{O}$
subtract$(\text{succ}(x), \text{succ}(y))$	$\equiv$	subtract(x, y)

function ack : number $\times$ number $\rightarrow$ number

ack$(\mathcal{O}, y)$	$\equiv$	succ(y)
ack$(\text{succ}(x), \mathcal{O})$	$\equiv$	ack$(x, \text{succ}(\mathcal{O}))$
ack$(\text{succ}(x), \text{succ}(y))$	$\equiv$	ack$(x, \text{ack}(\text{succ}(x), y))$

function foo : number $\times$ number $\rightarrow$ number

foo$(\mathcal{O}, \mathcal{O})$	$\equiv$	$\mathcal{O}$
foo$(\mathcal{O}, \text{succ}(\mathcal{O}))$	$\equiv$	foo$(\text{succ}(\mathcal{O}), \mathcal{O})$
foo$(\text{succ}(x), y)$	$\equiv$	foo$(x, \text{succ}(y))$

function f : number $\rightarrow$ number

f$(\mathcal{O})$	$\equiv$	$\mathcal{O}$
f$(\text{succ}(x))$	$\equiv$	succ$(\text{succ}(\text{f}(x)))$

function g : number $\rightarrow$ number

g$(\mathcal{O})$	$\equiv$	$\mathcal{O}$
g$(\text{succ}(x))$	$\equiv$	g$(\text{f}(x))$

function flatten : sexpr $\rightarrow$ sexpr

$$
\begin{array}{lcl}
\mathsf{flatten}(\mathsf{nil}) & \equiv & \mathsf{nil} \\
\mathsf{flatten}(\mathsf{atom}(n)) & \equiv & \mathsf{atom}(n) \\
\mathsf{flatten}(\mathsf{cons}(\mathsf{nil}, x)) & \equiv & \mathsf{cons}(\mathsf{nil}, \mathsf{flatten}(x)) \\
\mathsf{flatten}(\mathsf{cons}(\mathsf{atom}(n), x)) & \equiv & \mathsf{cons}(\mathsf{atom}(n), \mathsf{flatten}(x)) \\
\mathsf{flatten}(\mathsf{cons}(\mathsf{cons}(x, y), z)) & \equiv & \mathsf{flatten}(\mathsf{cons}(x, \mathsf{cons}(y, z)))
\end{array}
$$

function count : sexpr $\rightarrow$ number

$$
\begin{array}{lcl}
\mathsf{count}(\mathsf{nil}) & \equiv & \mathcal{O} \\
\mathsf{count}(\mathsf{atom}(n)) & \equiv & \mathsf{succ}(\mathcal{O}) \\
\mathsf{count}(\mathsf{cons}(\mathsf{nil}, x)) & \equiv & \mathsf{count}(x) \\
\mathsf{count}(\mathsf{cons}(\mathsf{atom}(n), x)) & \equiv & \mathsf{succ}(\mathsf{count}(x)) \\
\mathsf{count}(\mathsf{cons}(\mathsf{cons}(x, y), z)) & \equiv & \mathsf{count}(\mathsf{flatten}(\mathsf{cons}(\mathsf{cons}(x, y), z)))
\end{array}
$$

Führen Sie jeweils eine Terminierungsanalyse für das Programm P durch, das aus folgenden Datenstrukturen und Algorithmen besteht:

a) number, sexpr, flatten

b) number, sexpr, flatten, count

c) number, ack

d) number, subtract

e) number, foo

f) number, f, g

Geben Sie die zugehörige Präzedenz an, falls Sie für den Terminierungsbeweis die lexikographische Pfadordnung $\succ_{lpo}$ verwenden, und für nicht terminierende Algorithmen eine Eingabe, die eine unendliche Auswertung zur Folge hat. Für welche Algorithmen von P kann die Terminierung mit der Einbettungsordnung $\succ_{emb}$ gezeigt werden? Gibt es in P Algorithmen, die terminieren, deren Terminierung aber mit keiner lexikographischen Pfadordnung $\succ_{lpo}$ gezeigt werden kann?