Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Clemens Ballarin

Vorlesung Semantik
Wintersemester 2002/03
Übungsblatt 10
Besprechung am 16. Januar 2003

Aufgabe 1 (H) (*Funktionenräume*)

Seien $(C, \sqsubseteq_C)$ und $(D, \sqsubseteq_D)$ cpos. Der Funktionenraum $[C \to D]$ besteht aus der Menge der stetigen Funktionen von C nach D. Sei weiterhin die Ordnung $\sqsubseteq$ auf diesem Funktionenraum wie folgt definiert:

$$f \sqsubseteq g \iff \forall c \in C. f(c) \sqsubseteq_D g(c)$$

Die kleinste obere Schranke einer ω-Kette $(f_i)_{i\in\mathbb{N}}$ ergibt sich damit punktweise, d.h. zu $f_0 \sqsubseteq f_1 \sqsubseteq f_2 \sqsubseteq f_3 \ldots$ ist das Supremum $\bigsqcup f_i$ wie folgt definiert:

$$\bigsqcup_i f_i = \lambda x. \bigsqcup_i (f_i(x))$$

Zeigen Sie, dass $\bigsqcup_i f_i$ wohldefiniert ist (existiert die rechte Seite?), und dass es sich dabei tatsächlich um das Supremum handelt. Zeigen Sie weiterhin, dass das Supremum wieder ein Element von $[C \to D]$ ist, und damit $([C \to D], \sqsubseteq)$ eine cpo. Sie können dabei folgendes Vertauschungslemma benutzen:

Seinen $e_{n,m}$ $(n, m \in \mathbb{N})$ Elemente einer cpo $(E, \sqsubseteq_E)$ und gilt für alle $n \leq n'$ und $m \leq m'$, dass $e_{n,m} \sqsubseteq_E e_{n',m'}$, so gilt

$$\bigsqcup_i \bigsqcup_j e_{i,j} = \bigsqcup_j \bigsqcup_i e_{i,j}.$$

Aufgabe 2 (Ü) (*Stetigkeit von* Δ)

Sei die Funktion $\Delta : 2^\Sigma \to 2^\Sigma$ zur Terminierungsmenge einer Schleife **while** b **do** s definiert wie in der Vorlesung:

$$\Delta(M) = \{\sigma \mid \mathcal{B}[\![b]\!]_\sigma = tt \Rightarrow (\sigma \in \mathcal{T}[\![s]\!] \wedge \mathcal{S}[\![s]\!]_\sigma \subseteq M)\}$$

Zeigen Sie mit Hilfe der Aussage

$$\sigma \in \mathcal{T}[\![s]\!] \Rightarrow |\mathcal{S}[\![s]\!]_\sigma| < \infty$$

die Stetigkeit von Δ.

Aufgabe 3 (Ü) (*Sprünge in WHILE*)

Die Sprache WHILE wird um Markendeklarationen und Sprünge erweitert. Die Menge **Stmt** der Anweisungen ist definiert durch:

$$s ::= \textbf{skip} \mid X := a \mid s_0 \,;\, s_1 \mid \textbf{if } b \textbf{ then } s_0 \textbf{ else } s_1 \mid \textbf{while } b \textbf{ do } s \mid l : s \mid \textbf{goto } l$$

Dabei stammt l aus einer Menge **Lab** von Marken.

Für **Stmt** definieren wir eine denotationelle *Continuation*-Semantik. Sei die Menge $\mathbf{Cont} = \Sigma \to \Sigma$ definiert wie in der Vorlesung. Jeder Marke l, die in einem Programm definiert wird, wird eine *Continuation* zugeordnet, die das Programmverhalten ab dem definierenden Auftreten "$l:\ \ldots$" von l beschreibt. Alle solchen Zuordnungen sind in der Menge

$$\mathbf{Lenv} = \mathbf{Lab} \to \mathbf{Cont}$$

von Markenumgebungen enthalten.

a) Geben Sie eine Funktion

$$\mathcal{C} : \mathbf{Stmt} \to \mathbf{Lenv} \to \mathbf{Cont} \to \mathbf{Cont}$$

an mit folgender Bedeutung: Ist s eine Anweisung in einem Programm, e die durch das Programm festgelegte Markenumgebung und k die *Continuation* nach Ausführung von s, so gibt $\mathcal{C}[\![s]\!]e\ k$ das Verhalten von s im Kontext von e und k wieder.

b) Geben Sie eine Funktion

$$\mathcal{J} : \mathbf{Stmt} \to \mathbf{Lenv} \to \mathbf{Cont} \to \mathbf{Lenv}$$

an mit folgender Bedeutung: Ist s eine Anweisung, e eine Markenumgebung und k die *Continuation* nach Ausführung von s, so erweitert $\mathcal{J}[\![s]\!]e\ k$ die Markenumgebung e um alle in s definierten Marken. Ist eine Marke mehrmals in einem Programm definiert, so zählt das textuell letzte Auftreten. Stützen Sie die Definition von $\mathcal{J}$ auf die Funktion $\mathcal{C}$ aus Teilaufgabe (a).

c) Geben Sie eine Funktion

$$\mathcal{P} : \mathbf{Stmt} \to \mathbf{Cont}$$

an, die eine Anweisung als Programm interpretiert. D.h. einer Anweisung s wird ihr Zustandsübergangsverhalten unter der Markenumgebung zugeordnet, die in s definiert wird. Stützen Sie die Definition von $\mathcal{P}$ auf die Funktionen $\mathcal{C}$ und $\mathcal{J}$ aus den Teilaufgaben (a) und (b). Beschreiben Sie die durch s definierte Markenumgebung durch den kleinsten Fixpunktoperator.

Aufgabe 4 (P) (*Nichtdeterminismus*)

Erweitern Sie die denotationelle Semantik von WHILE in Gofer aus Aufgabe 4, Blatt 8, um das nichtdeterministische Statement

$$s_1 \ \mathbf{or}\ s_2$$

Ändern Sie ihr Programm dazu so, dass zu einem Statement und einem Zustand nicht nur ein Folgezustand berechnet wird, sondern die Liste aller möglichen Folgezustände. Testen Sie ihr Programm mit den beiden Anweisungen

$$\mathbf{while}\ \neg(x = 0)\ \mathbf{do}\ (x := 0\ \mathbf{or\ skip})$$

und

$$\mathbf{while}\ \neg(x = 0)\ \mathbf{do}\ (\mathbf{skip\ or}\ x := 0).$$