Logik, Semantik und Verifikation SS 2002: Musterlösung zum 8. Übungsblatt

Prof. Dr. Gert Smolka, Dipl.-Inform. Tim Priesnitz

Aufgabe 8.1: Grundregeln (5)

(a)

$$\begin{array}{rcl} R & = & \{\langle \emptyset, \langle 1, 1\rangle\rangle\} \\ & \cup & \{\langle\{\langle n, m\rangle\}, \langle n', m'\rangle\rangle \mid n, m \in \mathbb{N},\ n' = n+1,\ m' = m + 2n + 1\} \end{array}$$

(b)

$$\begin{array}{rcl} \hat{R}^0(\emptyset) & = & \emptyset \\ \hat{R}^1(\emptyset) & = & \{\langle 1, 1\rangle\} \\ \hat{R}^2(\emptyset) & = & \{\langle 1, 1\rangle, \langle 2, 4\rangle\} \\ \hat{R}^3(\emptyset) & = & \{\langle 1, 1\rangle, \langle 2, 4\rangle, \langle 3, 9\rangle\} \\ \hat{R}^4(\emptyset) & = & \{\langle 1, 1\rangle, \langle 2, 4\rangle, \langle 3, 9\rangle, \langle 4, 16\rangle\} \end{array}$$

Aufgabe 8.2: Grundregeln und Fixpunkte (8)

(a)

$$R = \{\langle \emptyset, 4\rangle\} \cup \{\langle\{x, y\}, z\rangle \mid x, y \in \mathbb{Z},\ z = x + y \mod 5\}$$

(b)

$$\begin{array}{rcll} \hat{R}^0(\emptyset) & = & \emptyset & \\ \hat{R}^1(\emptyset) & = & \{4\} & \\ \hat{R}^2(\emptyset) & = & \{3, 4\} & \\ \hat{R}^3(\emptyset) & = & \{1, 2, 3, 4\} & \\ \hat{R}^4(\emptyset) & = & \{0, 1, 2, 3, 4\} & \\ \hat{R}^n(\emptyset) & = & \{0, 1, 2, 3, 4\} & \quad n \geq 4 \end{array}$$

(c) $\bigcup_{i \in \mathbb{N}} \hat{R}^i(\emptyset) = \hat{R}^4(\emptyset) = \{0, 1, 2, 3, 4\}$

(d) Nach Fixpunktsatz (5.3.3) ist der kleinste Fixpunkt von $\hat{R}$ gerade $\bigcup_{i \in \mathbb{N}} \hat{R}^i(\emptyset)$.

(e) Wiederum gilt nach dem Fixpunktsatz, dass $P = \bigcup_{i \in \mathbb{N}} \hat{R}^i(\emptyset)$.

Aufgabe 8.3: Regeln mit unendlich vielen Prämissen (8) Wir definieren R als:

$$R := \{\langle \mathbb{N}, 0 \rangle\}$$

Jetzt betrachten wir die Kette $A_i := \{n \mid n \in \mathbb{N} \text{ und } n \leq i\}$. Dann gilt:

$$\hat{R}\left(\bigcup_{i \in \mathbb{N}} A_i\right) = \hat{R}(\mathbb{N}) = \{0\}$$

und

$$\bigcup_{i \in \mathbb{N}} \hat{R}(A_i) = \bigcup_{i \in \mathbb{N}} \emptyset = \emptyset$$

Also $\hat{R}\left(\bigcup_{i \in \mathbb{N}} A_i\right) \neq \bigcup_{i \in \mathbb{N}} \hat{R}(A_i)$. Damit ist $\hat{R}$ nicht stetig.

Aufgabe 8.4: Unstetige Funktionen (8) Seien die Funktionen $f_\perp$ und g und das Funktional F wie folgt definiert:

$$\begin{array}{rcll} f_\perp & = & \lambda n \in \mathbb{N}.\, \perp & \in \mathbb{N} \to \mathbb{N}_\perp \\ g & = & \lambda n \in \mathbb{N}.\, 1 & \in \mathbb{N} \to \mathbb{N}_\perp \\ & & & \\ F & = & \lambda f \in \mathbb{N} \to \mathbb{N}_\perp.\text{if } f = g \text{ then } f \text{ else } f_\perp & \end{array}$$

Es ist einfach zu sehen, dass F monoton ist. F ist aber nicht stetig, denn die aufsteigende Kette

$$f_i = \lambda n \in \mathbb{N}.\text{if } n \leq i \text{ then } 1 \text{ else } \perp$$

hat als kleinste obere Schranke

$$\bigsqcup_{i \in \mathbb{N}} f_i = g$$

und einerseits ist

$$F(\bigsqcup_{i \in \mathbb{N}} f_i) = F(g) = F$$

aber

$$\bigsqcup_{i \in \mathbb{N}} F(f_i) = \bigsqcup_{i \in \mathbb{N}} f_\perp = f_\perp \neq g$$

Also ist F nicht stetig.

Aufgabe 8.5: Approximation von Funktionen (6)

(a)
```
val maxFun = fn max => fn (n,m) => if n=0 then m
                                    else if m=0 then n
                                         else 1 + max(n-1,m-1)
```

(b)
```
exception Approx;
fun max0 (n:int,m:int) : int = raise Approx;
val max1 = maxFun max0;
val max2 = maxFun max1;
val max3 = maxFun max2;
val max4 = maxFun max3;
```

(c)
```
fun approx k = if k<1 then max0 else maxFun(approx(k-1))
```

Aufgabe 8.6: Denotationale Semantik für IMP (10)

(a)
```
fun da (Con n)      s = n
 |  da (Loc l)      s = s l
 |  da (Add(a1,a2)) s = da a1 s + da a2 s
```

(b)
```
fun db (LE(a1,a2)) s = da a1 s <= da a2 s
```

(c)
```
fun gamma (beta,phi) iter s = if beta s then iter (phi s) else s
```

(d)
```
fun dc (Assign(l,a)) s = (fn l' => if l=l' then da a s else s l')
 |  dc (Seq(c1,c2))  s = dc c2 (dc c1 s)
 |  dc (If(b,c1,c2)) s = if db b s then dc c1 s else dc c2 s
 |  dc (While(b,c))  s = fix (gamma (db b,dc c)) s
```

Aufgabe 8.7: Programmieren in IMP (5)

```
N := argument;
Y := 1;
while 2 <= N do (
   M := N; X := Y;
   while 2 <= M do (
      X := X+Y; M := M+ ~1);
   Y := X;
   N := N+ ~1);
result := Y
```