Theorie der Informatik

G. Röger
Frühjahrssemester 2019

Universität Basel
Fachbereich Informatik

Übungsblatt 12 — Lösungen

Aufgabe 12.1 (LOOP-Programme, 1 Punkt)
Welche Funktion berechnet folgendes Programm?

LOOP x_1 DO
  $x_1 := x_1 + 1$
END;
LOOP x_1 DO
  $x_1 := x_1 + 1$
END;
$x_0 := x_1$

Lösung:
$f(x) = 4x$

Aufgabe 12.2 (LOOP-Berechenbarkeit, 0.5 Punkte)
Betrachten Sie folgende Funktion g, die eine modifizierte Modulooperation berechnet:

$$g(x, y) = \begin{cases} x \bmod y, & \text{falls } y > 0 \\ \text{undefiniert}, & \text{sonst} \end{cases}$$

Ist g LOOP-berechenbar?

Lösung:
Nein, da nur totale Funktionen LOOP-berechenbar sein können.

Aufgabe 12.3 (Alternative Definition von LOOP-Programmen, 2 Punkte)
Zeigen Sie, dass wir mit der folgenden Definition von LOOP'-Programmen genau die gleichen Funktionen berechnen können wir mit der Definition von LOOP-Programmen aus der Vorlesung:

LOOP'-Programme sind induktiv wie folgt definiert:

- $x_i := x_j$ ist ein LOOP'-Programm für alle $i, j \in \mathbb{N}_0$ (*Zuweisung*)
- $x_i := x_i + 1$ ist ein LOOP'-Programm für alle $i \in \mathbb{N}_0$ (*Inkrementierung*)
- $x_i := x_i - 1$ ist ein LOOP'-Programm für alle $i \in \mathbb{N}_0$ (*Modifizierte Dekrementierung*)
- Sind P_1 und P_2 LOOP'-Programme, dann auch $P_1; P_2$ (*Komposition*)
- Ist P ein LOOP'-Program, dann auch
  LOOP x_i *DO* P *END*
  für alle $i \in \mathbb{N}_0$ (*LOOP-Schleife*)

Lösung:
Komposition und LOOP-Schleifen gibt es in beiden Definitionen, es reicht also, wenn wir zeigen, wie man Zuweisung, Inkrementierung und modifizierte Dekrementierung mit LOOP-Programmen (wie in der Vorlesung definiert) simulieren können und dass wir Addition und modifizierte Subtraktion mit LOOP'-Programmen simulieren können.

Wir haben in der Vorlesung die Zuweisung bereits als syntaktischen Zucker für LOOP-Programme eingeführt. Inkrementierung und Dekrementierung können mit Addition und modifizierter Subtraktion simuliert werden, indem man $i = j$ und $c = 1$ setzt.
Für die umgekehrte Richtung können wir einen LOOP-Programmausdruck $x_i := x_j + c$ mit dem folgenden LOOP'-Programm simulieren:

$x_i := x_j$;
$x_i := x_i + 1$;
$\vdots$ (insgesamt c mal)
$x_i := x_i + 1$;

Analog können wir c Dekrementierungsoperationen für die modifizierte Subtraktion $x_i := x_j - c$ verwenden.

Aufgabe 12.4 (Syntaktischer Zucker, 1.5 + 1.5 + 1.5 Punkte)
Geben Sie an, wie sich die folgenden syntaktischen Konstrukte für LOOP-Programme (mit der offensichtlichen Semantik) durch bekannte Konstrukte simulieren lassen. Sie dürfen dabei neben den Grundkonstrukten von LOOP-Programmen auch die zusätzlichen Konstrukte verwenden, die in Kapitel F1 eingeführt wurden.

(a) **IF** $x_i > c$ **THEN** P **ELSE** P' **END**

Lösung:

$x_k := x_i - c$;
IF $x_k \neq 0$ **THEN**
  P
END;
IF $x_k = 0$ **THEN**
  P'
END

Wobei x_k eine frische Variable ist.

(b) **IF** $x_i = x_j$ **THEN** P **END**

Lösung:

$x_k := x_i - x_j$;
$x_l := x_j - x_i$;
$x_m := x_k + x_l$;
IF $x_m = 0$ **THEN**
  P
END

Wobei x_k, x_l und x_m frische Variablen sind.

(c) **FOR** $x_i = 1$ **TO** c **DO** P **END**

Lösung:

$x_k := c$;
$x_i := 0$;
LOOP x_k **DO**
  $x_i := x_i + 1$;
  P
END

Wobei x_k eine frische Variable ist.

Aufgabe 12.5 (2 Punkte)

Geben sie ein LOOP-Programm an, welches die Potenzfunktion $f(x, y) = x^y$ berechnet. Sie dürfen allen syntaktischen Zucker aus der Vorlesung verwenden.

Lösung:

$x_0 := 1;$
LOOP x_2 DO
  $x_3 := 0;$
  LOOP x_1 DO
    $x_3 := x_3 + x_0$
  END;
  $x_0 := x_3$
END