12. Übung (9.–13. Juli 2018)

Programmierung

Aufgabe 1 (AGS 17.7)

Übersetzen Sie, wie aus der Vorlesung bekannt, folgendes C_0-Programm in ein H_0-Programm.

```
1  #include <stdio.h>
2
3  int main() {
4    int x1, x2;
5    scanf("%i", &x1);
6    x2 = 1;
7
8    while (x1 > 0) {
9      x2 = x2 * x1;
10     x1 = x1 - 1;
11   }
12   printf("%d", x2);
13   return 0;
14 }
```

Aufgabe 2 (AGS 17.25 a)

Übersetzen Sie folgende Funktion `h :: Int -> Int -> Int -> Int` eines H_0-Programms mittels *funtrans* in ein AM_0-Programm mit baumstrukturierten Adressen.

```
h x1 x2 x3 = if x3 > x1  then x2 - 1  else h x2 (x1 - x3) x2
```

Aufgabe 3 (AGS 17.27 c)

Das folgende H_0-Programmstück ist durch Nutzung der Transformationsfunktion aus der Vorlesung aus Statements eines entsprechenden C_0-Programms entstanden. Geben Sie diese an.

```
1  f1  x1 = if x1 `mod` 2 == 0
2           then f11 x1
3           else f12 x1
4  f11 x1 = f2 (x1 `div` 2)
5  f12 x1 = f2 (x1 - 1)
6  f2  x1 = f3 (2 * x1)
```

Aufgabe 4 (AGS 12.3.8 ⋆)

```
1  data Tree = Leaf Int | Branch Int Tree Tree
2
3  sumTree :: Tree -> Int
4  sumTree (Leaf i) = i
5  sumTree (Branch i t1 t2) = i + sumTree t1 + sumTree t2
6
7  revTree :: Tree -> Tree
8  revTree (Leaf i) = Leaf i
9  revTree (Branch i t1 t2) = Branch i (revTree t2) (revTree t1)
10
11 overlay :: Tree -> Tree -> Tree
12 overlay (Leaf i1) (Leaf i2) = Leaf (i1 + i2)
13 overlay (Branch i1 t11 t12) (Branch i2 t21 t22)
14   = Branch (i1 + i2) (overlay t11 t21) (overlay t12 t22)
```

Zeigen Sie durch strukturelle Induktion und unter Verwendung der obigen Definitionen, dass folgende Aussage gilt:

$$\forall \texttt{t :: Tree} : \quad \texttt{sumTree (overlay t t)} = 2 \texttt{ * (sumTree (revTree t))}$$

Zusatzaufgabe 1 (AGS 16.28)

(a) Transformieren Sie die folgende Funktion `f` eines H_0-Programms in ein AM_0-Programm. Sie können dabei die Adressen frei vergeben, sie müssen nicht baumstrukturiert sein. Geben Sie keine Zwischenschritte an.

```
f :: Int -> Int -> Int
f x1 x2 = if x1 <= 0 then x2 else f (x1 / 2) (x2 + x1)
```

(b) Das folgende H_0-Programmstück ist durch Nutzung der Transformationsfunktion aus der Vorlesung aus Statements eines entsprechenden C_0-Programms entstanden. Geben Sie diese Statements an.

```
f1    x1 x2 = if x1 > 0  then f11 x1 x2  else f2  x1 x2
f11   x1 x2 = f111 x1 x2
f111  x1 x2 = if x1 `mod` 2 == 0  then f1111 x1 x2
                                  else f112  x1 x2
f1111 x1 x2 = f112 x1 (x2 + x1)
f112  x1 x2 = f1 (x1 - 1) x2
```