

Exercise Sheet 1

Sample Solution

Task 1: Recursion and Structural Induction

(a)

$$\begin{aligned}
 z[x := a'] &= z \\
 y[x := a'] &= \begin{cases} a', & \text{if } x = y \\ y, & \text{if } x \neq y \end{cases} & y \in \text{Var} \\
 (a_1 \oplus a_2)[x := a'] &= a_1[x := a'] \oplus a_2[x := a'] \quad \text{for } \oplus \in \{+, -, *\}
 \end{aligned}$$

(b)

$$\text{occ} : \text{AExp} \times \text{Var} \rightarrow \mathbb{N}$$

$$\begin{aligned}
 \text{occ}(z, x) &= 0 \\
 \text{occ}(y, x) &= \begin{cases} 1, & \text{if } x = y \\ 0, & \text{if } x \neq y \end{cases} & y \in \text{Var} \\
 \text{occ}(a_1 \oplus a_2, x) &= \text{occ}(a_1, x) + \text{occ}(a_2, x) \quad \text{for } \oplus \in \{+, -, *\}
 \end{aligned}$$

(c)

The statement $\mathbf{FV}(a[x := a']) \subseteq (\mathbf{FV}(a) \setminus \{x\}) \cup \mathbf{FV}(a')$ will be shown by structural induction on the structure of arithmetic expressions a .

Induction Base:

Case 1: $a = z, z \in \mathbb{Z}$

$$\begin{aligned}
 \mathbf{FV}(z[x := a']) &\stackrel{(a)}{=} \underbrace{\mathbf{FV}(z)}_{=\emptyset} = \mathbf{FV}(z) \setminus \{x\} \subseteq (\mathbf{FV}(z) \setminus \{x\}) \cup \\
 &\mathbf{FV}(a')
 \end{aligned}$$

Case 2: $a = y, y \neq x \quad y \in \text{Var}$

$$\mathbf{FV}(y[x := a']) \stackrel{(a)}{=} \underbrace{\mathbf{FV}(y)}_{=\{y\}} \subseteq (\mathbf{FV}(y) \setminus \{x\}) \cup \mathbf{FV}(a')$$

Case 3: $a = x$

$$\mathbf{FV}(x[x := a']) \stackrel{(a)}{=} \mathbf{FV}(a') \subseteq (\mathbf{FV}(y) \setminus \{x\}) \cup \mathbf{FV}(a')$$

Induction Hypothesis:

Assume for all subexpressions $\bar{a}$ of a that
 $\mathbf{FV}(\bar{a}[x := a']) \subseteq (\mathbf{FV}(\bar{a}) \setminus \{x\}) \cup \mathbf{FV}(a')$ holds.

Induction Step:

Let $\oplus \in \{+, -, *\}$ and $a_1, a_2 \in \text{AExp}$.

$$\begin{aligned} \mathbf{FV}((a_1 \oplus a_2)[x := a']) &\stackrel{(a)}{=} \mathbf{FV}(a_1[x := a_1] \oplus a_2[x := a']) \\ &\stackrel{(\text{Def 2.4})}{=} \mathbf{FV}(a_1[x := a']) \cup \mathbf{FV}(a_2[x := a']) \\ &\stackrel{\text{I.H.}}{\subseteq} (\mathbf{FV}(a_1) \setminus \{x\}) \cup \mathbf{FV}(a') \cup (\mathbf{FV}(a_2) \setminus \{x\}) \cup \mathbf{FV}(a') \\ &= (\mathbf{FV}(a_1) \cup \mathbf{FV}(a_2)) \setminus \{x\} \cup \mathbf{FV}(a') \\ &\stackrel{(\text{Def 2.4})}{=} (\mathbf{FV}(a_1 \oplus a_2)) \setminus \{x\} \cup \mathbf{FV}(a') \end{aligned}$$

□

(d)

(i)

$$\text{length}(a[x := a']) = \text{length}(a) + \text{occ}(a, x) \cdot (\text{length}(a') - 1)$$

(ii)

The correctness of the proposed formula will be proven by structural induction on the structure of arithmetic expressions a .

Induction Base:

Case 1: $a = z, z \in \mathbb{Z}$

$$\begin{aligned} \text{length}(z[x := a']) &\stackrel{(a)}{=} \text{length}(z) \\ &= \text{length}(z) + 0 \cdot (\text{length}(a') - 1) \\ &\stackrel{(b)}{=} \text{length}(z) + \text{occ}(z, x) \cdot (\text{length}(a') - 1) \end{aligned}$$

Case 2: $a = y, y \neq x, y \in \text{Var}$

$$\begin{aligned} \text{length}(y[x := a']) &\stackrel{(a)}{=} \text{length}(y) \\ &= \text{length}(y) + 0 \cdot (\text{length}(a') - 1) \\ &\stackrel{(b)}{=} \text{length}(y) + \text{occ}(y, x) \cdot (\text{length}(a') - 1) \end{aligned}$$

Case 3: $a = x$

$$\begin{aligned}
\text{length}(x[x := a']) &\stackrel{(a)}{=} \text{length}(a') \\
&= 1 + \text{length}(a') - 1 \\
&= 1 + 1 \cdot (\text{length}(a') - 1) \\
&\stackrel{(b)}{=} \text{length}(x) + \text{occ}(x, x) \cdot (\text{length}(a') - 1)
\end{aligned}$$

Induction Hypothesis:

Assume for all subexpressions $\bar{a}$ of a that
 $\text{length}(\bar{a}[x := a']) = \text{length}(\bar{a}) + \text{occ}(\bar{a}, x) \cdot (\text{length}(a') - 1)$ holds.

Induction Step:

Let $\oplus \in \{+, -, *\}$ and $a_1, a_2 \in \text{AExp}$.

$$\begin{aligned}
\text{length}((a_1 \oplus a_2)[x := a']) &\stackrel{(a)}{=} \text{length}(a_1[x := a'] \oplus a_2[x := a']) \\
&\stackrel{\text{def length}}{=} 1 + \text{length}(a_1[x := a']) + \text{length}(a_2[x := a']) \\
&\stackrel{\text{I.H.}}{=} 1 + (\text{length}(a_1) + \text{occ}(a_1, x) \cdot (\text{length}(a') - 1)) + \text{length}(a_2[x := a']) \\
&\stackrel{\text{I.H.}}{=} 1 + (\text{length}(a_1) + \text{occ}(a_1, x) \cdot (\text{length}(a') - 1)) \\
&\quad + (\text{length}(a_2) + \text{occ}(a_2, x) \cdot (\text{length}(a') - 1)) \\
&= (1 + \text{length}(a_1) + \text{length}(a_2)) \\
&\quad + (\text{occ}(a_1, x) + \text{occ}(a_2, x)) \cdot (\text{length}(a') - 1) \\
&\stackrel{\text{Def}}{=} \text{length}(a_1 \oplus a_2) + \text{occ}(a_1 \oplus a_2, x) \cdot (\text{length}(a') - 1)
\end{aligned}$$

□

Task 2: The Programming Language WHILE

(a)

```

x := 1;      // f0
y := 1;      // f1
i := 1;
while (i < n) do
    t := x + y;    // fn+2
    x := y;        // fn+1
    y := t;        // fn+2
    i := i + 1

```

end

(b)

