Exercise 1. Theory of Programming Languages

JAVIER SAGREDO TAMAYO

October 15, 2021

Exercise 1. 1. We will define the operation in a compositional way:

$$\_[\_/\_] : \mathbf{AExp} \times \mathbf{Var} \times \mathbf{AExp} \to \mathbf{AExp}$$

$$\begin{array}{rcl} e[x/e'] & := & n[x/e'] := n \\ & | & x[x/e'] := e' \\ & | & y[x/e'] := y \\ & | & e_1[x/e'] + e_2[x/e'] \\ & | & e_1[x/e'] - e_2[x/e'] \\ & | & e_1[x/e'] * e_2[x/e'] \end{array}$$

where n is a constant, $x, y \in \mathbf{Var}$, $x \neq y$ and $e, e', e_1, e_2 \in \mathbf{AExp}$.

2. In order to prove the lemma, we are going to use structural induction. First we will prove the lemma for the base cases, and then assuming the induction hypothesis we will prove the composite cases.

   Base cases:

   - **constant:** The case when $e = n \in \mathbb{Z}$ is trivial following from the definitions of $\mathcal{A}[\![n]\!]\sigma$ and $\_[\_/\_]$. Note that $n = \mathcal{A}[\![n]\!]\sigma'$ for any state $\sigma' \in \mathbf{State}$.

     $$\mathcal{A}[\![n[x/e']]\!]\sigma = \mathcal{A}[\![n]\!]\sigma = n = \mathcal{A}[\![n]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma]$$

   - **variable:**
     - $e = x$: The case where $e = x \in \mathbf{Var}$ follows from the definition of $\_[\_/\_]$ and the fact that mapping a variable to a specific value and then requesting the value for that same variable on the modified state yields the same value we used as an input, or more precisely:

       $$\sigma[v \mapsto n](v) = n, \forall v \in \mathbf{Var}, n \in \mathbb{Z}, \sigma \in \mathbf{State} \qquad (1)$$

       So then we can prove:

       $$\mathcal{A}[\![x[x/e']]\!]\sigma = \mathcal{A}[\![e']\!]\sigma \overset{(1)}{=} \sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma](x) = \mathcal{A}[\![x]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma]$$

– $e = y \neq x$: The case when $e = y \neq x$, where $x, y \in \mathbf{Var}$ is trivial following from the definition of $\_[\_/\_]$ and from the fact that altering a value on the state that is not used in the expression evaluated by $\mathcal{A}$, the result of the evaluation doesn't change, or more precisely:

$$\mathcal{A}[\![y]\!]\sigma = \mathcal{A}[\![y]\!]\sigma', \forall \sigma, \sigma' \in \mathbf{State} \text{ such that } \sigma(y) = \sigma'(y) \quad (2)$$

So then we can prove the following:

$$\mathcal{A}[\![y[x/e']]\!]\sigma = \mathcal{A}[\![y]\!]\sigma = \sigma(y) \overset{(2)}{=} \sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma](y) = \mathcal{A}[\![y]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma]$$

Induction Hypothesis (I.H.): If $e = e_1 \oplus e_2$, where $e_1, e_2 \in \mathbf{AExp}$ and $\oplus$ is one of $+, -, *$, then we are going to assume that:

$$\mathcal{A}[\![e_1[x/e']]\!]\sigma = \mathcal{A}[\![e_1]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma]$$

and

$$\mathcal{A}[\![e_2[x/e']]\!]\sigma = \mathcal{A}[\![e_2]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma]$$

Composite cases: These cases (equivalent regardless of the specific operation involved) follow directly from the definition of $\mathcal{A}$ and $\_[\_/\_]$ combined with the induction hypothesis.

$$\begin{aligned}
\mathcal{A}[\![(e_1 \oplus e_2)[x/e']]\!]\sigma &= \mathcal{A}[\![e_1[x/e'] \oplus e_2[x/e']]\!]\sigma \\
&= \mathcal{A}[\![e_1[x/e']]\!]\sigma \oplus_{\mathbb{Z}} \mathcal{A}[\![e_2[x/e']]\!]\sigma \\
&\overset{(I.H.)}{=} \mathcal{A}[\![e_1]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma] \oplus_{\mathbb{Z}} \mathcal{A}[\![e_2]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma] \\
&= \mathcal{A}[\![e_1 \oplus e_2]\!]\sigma[x \mapsto \mathcal{A}[\![e']\!]\sigma]
\end{aligned}$$

□