CS 520 :: Problem Set 7 Solutions

1. calculator \[f\] =
   \[\text{acc} = 0, \ \text{arg} = 0,\]
   \[\text{sign} = 1,\]
   \[\text{enter} = \zeta(s)A(n)\text{.arg} := n,\]
   \[\text{add} = \zeta(s)A(s,\text{acc} := s.\text{equals})\text{.sign} := 1,\]
   \[\text{sub} = \zeta(s)A(s,\text{acc} := s.\text{equals})\text{.sign} := -1,\]
   \[\text{equals} = \zeta(s)A\text{.acc} + s.\text{sign} \ast s.\text{arg}\]

2. \[\text{apply} = f \text{.lambda} f(x)\].
   \[\langle \text{apply} \rangle = \{\text{arg} = \zeta(f)\text{.f.arg, \ val} = \zeta(f)\text{.f} \text{.x.arg, \ val} = \zeta(x)\langle f \text{.x} \rangle\}\{f \leftarrow f.\text{arg}\}\]
   \[\equiv [\text{arg} = \zeta(f)\text{.f.arg, \ val} = \zeta(f)\langle \text{arg} = \zeta(x)\text{.x.arg, \ val} = \zeta(x)\langle f \text{.x} \rangle\}\{f \leftarrow f.\text{arg}\}\]
   \[\equiv [\text{arg} = \zeta(f)\text{.f.arg, \ val} = \zeta(f)\langle \text{arg} = \zeta(x)\text{.x.arg, \ val} = \zeta(x)\langle f \text{.x} \rangle\}\{f \leftarrow f.\text{arg}\}\]

(b) \[\langle \lambda(x)\rangle = \{\text{arg} = \zeta(x)\text{.x.arg, \ val} = \zeta(x)\text{.x.arg}\].
   \[\langle \text{apply} \rangle = \{\text{arg} = \zeta(x)\text{.x.arg, \ val} = \zeta(x)\text{.x.arg}\]

   The base case is when the term \(c\) is an object (has the shape of an object). \(\text{Outcome}(c)\) will return \(c\) itself, which is consistent with the (Red Object) rule of the operational semantics. For the
   inductive steps, the assumption will be that a subterm of the term passed to \(\text{Outcome}\) is evaluated consistently with the rules of the operational semantics; that is, if \(\text{Outcome}(a) = o\), then \(a \rightarrow o\),

- Case 1. Selection: \(c \equiv a.l\). \(\text{Outcome}(c)\) first evaluates \(a\) recursively. If the result \(o\) is an
   object, it proceeds. This first check corresponds to one of the preconditions of the (Red Select)
   rule, \(a \rightarrow v\), where \(v\) is an object. The condition is satisfied by the inductive hypothesis.
   \(\text{Outcome}\) also checks whether the attribute index \(j\) is valid for the object \(o\), and so does (Red
   Select). With these conditions satisfied, \(\text{Outcome}\) performs the substitution and (recursively)
   evaluates the result to an object, which corresponds to the “conclusion” of (Red Select).
• Case 2. Method update: \( c \equiv a_j \Leftarrow \varsigma(x)b \). Similarly to the previous case, both \textit{Outcome} and (Red Update) require the subterm `a` to evaluate to an object and also validate the index \( j \) against that object. By the inductive hypothesis, \( o = \text{Outcome}(a) \) will be equivalent to the object that (Red Update) requires in its precondition. After the conditions are satisfied, \textit{Outcome} just replaces the method definition in `o` with the given one, while preserving all other methods, which is exactly the same as the conclusion of the (Red Update Rule).

General conclusion: Algorithm \textit{Outcome} is sound with respect to the operational semantics.

4. (a) \( a = \text{def} \ [l = \varsigma(x)[a = [[]].a] \); \( b = \text{def} \ [l = \varsigma(x)[]] \)

\( a \rightarrow b \), but not \( a \Rightarrow b \), because \( \Rightarrow \) is top-level reduction only.

(b) \( a_0 = \text{def} \ [l = \varsigma(s)[k = s.l]].l \)
\( a_0 \rightarrow [k = [l = \varsigma(s)[k = s.l]].l] \rightarrow \ldots \)

(c) \( a = \text{def} \ [l = \varsigma(x)a_0].l = \varsigma(x)[] \)

The \( a_0 \) subterm diverges, but if the method update is performed first, the reduction converges to \( [l = \varsigma(x)[]] \).

5. \( \Rightarrow \) does not operate within an object, but \( \rightarrow \) is allowed to do that.

\( a = \text{def} \ [l = \varsigma(x)[a := [[]].a] \)
\( a \rightarrow a \) (and no other reduction possible); \( a \rightarrow [l = \varsigma(x)[]] \).