COMP 523: Language-based security

Brigitte Pientka
Winter 2008

Exercise 2

The big-step operational semantics we have discussed so far is not very efficient in the sense that it re-evaluates expressions although they are already values. Consider for example the expression: (lam x. (s x)) (s (s z)). *Using the rules of the big-step semantics the expression* (s (s z)) *will be evaluated twice.*

Explore an alternative operational semantics in which expressions that are known to be values (since they have been evaluated) are not evaluated again. State and prove in which way the new semantics is equivalent to old one.

Hint*: It may be necessary to extend the language of expressions or explicitly separate the language of values from the language of expressions).*

Solution

We solve this problem in two steps: First, we define the new semantics in which any expression which has already been evaluated is marked by a new constructor vl , and second we show that the new and the original semantics are equivalent. Before we actually define the new semantics, let us clarify the equivalence relation between new and original semantics.

As a first approximation, we might say two semantics are equivalent if any expression e which evaluates to some value v in the original semantics evaluates to the same value v in the new semantics (and vice versa). However, the expressions of the original semantics and the new semantics need not be the same. Similarly, the definition of values in the original and new semantics need not be identical.

For example, consider the evaluation of the following expression in the new semantics: (**lam** x.**lam** $y.y$ x) **z** The return value will be **lam** $y.y$ (vl **z**) where **z** is marked by vl indicating that **z** is already a value. In the original semantics, (**lam** x.**lam** $y.y$ x) **z** evaluates to **lam** $y.y$ **z**. Clearly the value **lam** $y.y$ (vl **z**) is not identical to the value **lam** $y.y$ **z**. Moreover, the evaluation of (**lam** x.**lam** $y.y$ x) **z** might be part of a larger computation of an expression like the following:

$$(\mathbf{lam}\ x.\mathbf{lam}\ y.y\ x)\ \mathbf{z}\ (\mathbf{lam}\ w.w)$$

Therefore intermediate steps of this evaluation require us to evaluate expressions where vl v occurs as a subexpression. This example illustrates that the relationship between expressions and values of the original and new semantics is not trivial.

Therefore, we generalize our notion of equivalence. Two semantics are equivalent if evaluating an expression in the new semantics simulates the evaluation in the original semantics, and vice versa. This technique is called *bisimulation* and is illustrated in the following commutative diagram.

$$\begin{tikzcd}
e \arrow[r, "\mathcal{D} : e \hookrightarrow v"] \arrow[d, "\mathcal{C} : e \overset{e}{\rightsquigarrow} e'"'] & v \\
F \arrow[r, "\mathcal{E} : e' \overset{v}{\hookrightarrow} v'"'] & W \arrow[u, "\mathcal{C}' : v \overset{v}{\rightsquigarrow} v'"']
\end{tikzcd}$$

Fig. 1. Bisimulation

$\mathcal{D}$ represents the evaluation in the original semantics, while $\mathcal{E}$ denotes the evaluation in the new semantics. The relationship between expressions of the original and new semantics is described explicitly by $e \overset{e}{\rightsquigarrow} e'$. The relation $v \overset{v}{\rightsquigarrow} v'$ represents the correspondence between values in the original and new semantics.

To show soundness (see figure 2), we assume that we have an evaluation $\mathcal{E} : e' \overset{v}{\hookrightarrow} v'$ in the new semantics and a relation $\mathcal{C}$ between expressions e' of the new semantics and expressions e of the original semantics. Given this evaluation $\mathcal{E}$ and the relation $\mathcal{C}$, we construct a derivation for $\mathcal{D} : e \hookrightarrow v$ in the original semantics and a conversion $\mathcal{C}'$ of v to v'.

$$\begin{tikzcd}
e \arrow[r, dotted, "\mathcal{D} : e \hookrightarrow v"] \arrow[d, "\mathcal{C} : e \overset{e}{\rightsquigarrow} e'"'] & v \\
F \arrow[r, "\mathcal{E} : e' \overset{v}{\hookrightarrow} v'"'] & W \arrow[u, dotted, "\mathcal{C}' : v \overset{v}{\rightsquigarrow} v'"']
\end{tikzcd}$$

Fig. 2. Soundness

Theorem 1 (Soundness). *If $\mathcal{E} : e' \overset{v}{\hookrightarrow} v'$ and $\mathcal{C} : e \overset{e}{\rightsquigarrow} e'$ then there exists an evaluation $\mathcal{D} : e \hookrightarrow v$ and $\mathcal{C}' : v \overset{v}{\rightsquigarrow} v'$.*

For completeness (see figure 3), we prove the opposite direction. Given an evaluation $\mathcal{D} : e \hookrightarrow v$ in the original semantics and relation $\mathcal{C}$ between e and e', we construct a derivation $\mathcal{E} : e' \overset{v}{\hookrightarrow} v'$ and v' corresponds to v.

$$\begin{tikzcd}
e \arrow[r, "\mathcal{D} : e \hookrightarrow v"] \arrow[d, swap, "\mathcal{C} : e \overset{e}{\rightsquigarrow} e'"] & v \\
F \arrow[r, dotted, swap, "\mathcal{E} : e' \overset{v}{\hookrightarrow} v'"] & W \arrow[u, dotted, swap, "\mathcal{C}' : v \overset{v}{\rightsquigarrow} v'"]
\end{tikzcd}$$

Fig. 3. Completeness

Theorem 2 (Completeness). *If $\mathcal{D} : e \hookrightarrow v$ and $\mathcal{C} : e \overset{e}{\rightsquigarrow} e'$ then there exists an evaluation $\mathcal{E} : e' \overset{v}{\hookrightarrow} v'$ and $\mathcal{C}' : v \overset{v}{\rightsquigarrow} v'$.*

Expressions and Values in the new semantics

We start with the definition for expressions. To avoid confusion between expressions of the original and new semantics, we distinguish between them syntactically.

Expressions

$$e' ::= \mathsf{z} \mid \mathsf{s}\ e' \mid \mathsf{case}\ e'_1\ \mathsf{of}\ \mathsf{z}^* \Rightarrow e'_2 \mid \mathsf{s}^*\ x \Rightarrow e'_3 \mid \mathsf{lam}\ x.e' \mid e'_1 e'_2 \mid \mathsf{vl}\ v' \mid \ldots$$

The next question which arises is whether we want to distinguish between values and expression in the new semantics. Recall that in the original semantics we introduced a judgement to identify expressions which are values, but we did not separate expressions and values into two different syntactic categories. In the new semantics it is useful (but not necessary) to distinguish between expressions and values. One reason to enforce this distinction is, that the additional structure will give us more guidance during the development of the new semantics. Therefore we define values separately from expressions.

Values

$$v' ::= \mathsf{z}^* \mid \mathsf{s}^*\ v' \mid \mathsf{lam}^*\ x.e' \mid \ldots$$

Next, we need to decide how expressions and values fit together. Clearly, the constructor vl should take values as arguments and coerce values to expressions.

The more interesting question is whether we want to make the relationship between expressions and values in other expressions like $\mathsf{lam}\ x.e$ more explicit. In fact our intention is to substitute a value for the bound variable x in the body e of the function $\mathsf{lam}\ x.e$ and not arbitrary expressions. Similarly, when evaluating the case-statement $\mathsf{case}\ e_1'\ \mathsf{of}\ \mathsf{z}^* \Rightarrow e_2' \,|\, \mathsf{s}^*\ x \Rightarrow e_3'$, we substitute a value for x in e_3'. Therefore, we require any bound variable to stand for a value. To represent the identity function for example, we write $\mathsf{lam}\ x.(\mathsf{vl}\ x)$.

Evaluation rules

The judgement $e' \overset{v}{\hookrightarrow} v'$ takes expressions e' and returns their value v'. We only present a few evaluation rules to illustrate the main principle.

$$\frac{}{\mathsf{z} \overset{v}{\hookrightarrow} \mathsf{z}^*}\ \mathsf{ev\_z}_v \qquad \frac{e' \overset{v}{\hookrightarrow} v'}{\mathsf{s}\ e' \overset{v}{\hookrightarrow} \mathsf{s}^*\ v'}\ \mathsf{ev\_s}_v$$

$$\frac{e_1 \overset{v}{\hookrightarrow} \mathsf{z}^* \qquad e_2 \overset{v}{\hookrightarrow} v}{\mathsf{case}\ e_1\ \mathsf{of}\ \mathsf{z}^* \Rightarrow e_2 \,|\, \mathsf{s}^*\ x \Rightarrow e_3 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_case\_z}_v \qquad \frac{e_1 \overset{v}{\hookrightarrow} \mathsf{s}^*\ v_2 \qquad [v_2/x]e_3 \overset{v}{\hookrightarrow} v}{\mathsf{case}\ e_1\ \mathsf{of}\ \mathsf{z}^* \Rightarrow e_2 \,|\, \mathsf{s}^*\ x \Rightarrow e_3 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_case\_s}_v$$

$$\frac{}{\mathsf{lam}\ x.e \overset{v}{\hookrightarrow} \mathsf{lam}^*\ x.e}\ \mathsf{ev\_lam}_v$$

$$\frac{e_1 \overset{v}{\hookrightarrow} \mathsf{lam}^*\ x.e' \qquad e_2 \overset{v}{\hookrightarrow} v_2 \qquad [v_2/x]e' \overset{v}{\hookrightarrow} v}{e_1 e_2 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_app}_v$$

$$\frac{e_1 \overset{v}{\hookrightarrow} v_1 \qquad [v_1/x]e_2 \overset{v}{\hookrightarrow} v}{\mathsf{let\ val}\ x = e_1\ \mathsf{in}\ e_2 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_letv}_v$$

Conversions

Next, we explicitly define the relation of expressions in the original and new semantics. The interesting case is the relation between functions in both semantics. Let us recall again the previous example. The expression $\mathsf{lam}\ y.y\ (\mathsf{vl}\ \mathsf{z})$ in the new semantics should be related to $\mathbf{lam}\ y.y\ \mathbf{z}$ in the original semantics. This example illustrates two important aspects of the conversion: First, we need to convert the function body. Second, the bound variable y in $\mathsf{lam}\ y.y\ (\mathsf{vl}\ \mathsf{z})$ represents a value, while the bound variable y in $\mathbf{lam}\ y.y\ \mathbf{z}$ denotes an expression. Let us rename the variable y to x in $\mathbf{lam}\ y.y\ \mathbf{z}$ to avoid confusion. To convert $\mathbf{lam}\ x.x\ \mathbf{z}$ to $\mathsf{lam}\ y.y\ (\mathsf{vl}\ \mathsf{z})$, we convert $x\ \mathbf{z}$ to $y\ (\mathsf{vl}\ \mathsf{z})$ under the assumption that the expression x converts to the value y where x and y are new parameters. We can represent this idea using hypothetical and parametric judgements. The inference rule $\mathsf{c\_lam}$ is parametric in x and y and hypothetical in $u : x \overset{v}{\rightsquigarrow} y$.

$$\frac{}{\mathbf{z} \overset{e}{\rightsquigarrow} \mathsf{z}}\ \mathsf{c\_z} \qquad\qquad \frac{e \overset{e}{\rightsquigarrow} e'}{\mathbf{s}\ e \overset{e}{\rightsquigarrow} \mathsf{s}\ e'}\ \mathsf{c\_s}$$

$$\frac{e_1 \overset{e}{\rightsquigarrow} {e_1}' \qquad e_2 \overset{e}{\rightsquigarrow} {e_2}'}{e_1\ e_2 \overset{e}{\rightsquigarrow} e_1'\ e_2'}\ \mathsf{c\_app} \qquad\qquad \frac{v \overset{v}{\rightsquigarrow} v'}{v \overset{e}{\rightsquigarrow} \mathsf{vl}\ v'}\ \mathsf{c\_vl}$$

$$\frac{e_1 \overset{e}{\rightsquigarrow} {e_1}' \qquad e_2 \overset{e}{\rightsquigarrow} {e_2}' \qquad \begin{array}{c} \overline{x \overset{v}{\rightsquigarrow} y}\ u \\ \vdots \\ e_3 \overset{e}{\rightsquigarrow} {e_3}' \end{array}}{\mathbf{case}\ e_1\ \mathbf{of}\ \mathbf{z} \Rightarrow e_2 \,|\, \mathbf{s}\ x \Rightarrow e_3 \overset{e}{\rightsquigarrow} \mathsf{case}\ e_1'\ \mathsf{of}\ \mathsf{z}^* \Rightarrow e_2' \,|\, \mathsf{s}^*\ y \Rightarrow e_3'}\ \mathsf{c\_case\_z}$$

$$\frac{\begin{array}{c} \overline{x \overset{v}{\rightsquigarrow} y}\ u \\ \vdots \\ e \overset{e}{\rightsquigarrow} e' \end{array}}{\mathbf{lam}\ x.e \overset{e}{\rightsquigarrow} \mathsf{lam}\ y.e'}\ \mathsf{c\_lam}^{x,y,u}$$

Note that the inference system is non-deterministic. In the case where e is also a value two rules apply. For example, to convert $\mathbf{s}\ e$ we can apply the $\mathsf{c\_s}$ and th $\mathsf{c\_vl}$ rule. We also note that this conversion between expressions of the original and new semantics can be used in both directions, i.e. to translate expressions of the original semantics to the expressions in the new semantics (and vice versa).

The conversions between values of the original and new semantics follows the same principle as above.

$$\frac{}{\mathbf{z} \overset{v}{\rightsquigarrow} \mathsf{z}^*}\ \mathsf{cv\_z} \qquad\qquad \frac{e \overset{v}{\rightsquigarrow} v'}{\mathbf{s}\ e \overset{v}{\rightsquigarrow} \mathsf{s}\ v'}\ \mathsf{cv\_s}$$

$$\frac{\begin{array}{c} \overline{x \overset{v}{\rightsquigarrow} y}\ u \\ \vdots \\ e \overset{e}{\rightsquigarrow} e' \end{array}}{\mathbf{lam}\ x.e \overset{v}{\rightsquigarrow} \mathsf{lam}\ y.e'}\ \mathsf{cv\_lam}^{x,y,u}$$

Completeness Proof

After carefully designing the evaluation and conversion rules, we prove equivalence of the new and the original semantics. We focus on the completeness proof.

Theorem 3. *If $\mathcal{D} : e \hookrightarrow v$ and $\mathcal{C} : e \overset{e}{\leadsto} e'$ then $\mathcal{E} : e' \overset{v}{\hookrightarrow} v'$ and $\mathcal{C}' : v \overset{v}{\leadsto} v'$.*

The proof is by induction on the structure of $\mathcal{D}$. We will consider all possible cases for $\mathcal{D}$, and apply inversion on $\mathcal{C}$. As pointed out earlier, inversion on $e \overset{e}{\leadsto} e'$ is not unique. Consider the following case for $\mathsf{ev\_s}$ rule.

$$\mathcal{D} = \cfrac{\begin{array}{c}\mathcal{D}_1\\ e \hookrightarrow v\end{array}}{\mathbf{s}\ e' \hookrightarrow v}\ \mathsf{ev\_s}$$

By assumption we know $\mathcal{C} : \mathbf{s}\ e \overset{e}{\leadsto} \mathsf{s}\ e'$. Note that two possible rules apply $\mathsf{c\_vl}$ and $\mathsf{c\_s}$ depending on whether $\mathbf{s}\ e$ is already a value or not. This phenomena will occur for any expression which is also a value and we have one general case to cover this possibility.

$$\mathcal{D} = e \hookrightarrow v \qquad\qquad \mathcal{C} : \cfrac{e \overset{v}{\leadsto} v'}{e \overset{e}{\leadsto} \mathsf{vl}\ v'}\ \mathsf{c\_vl}$$

Note, that we will not be able to prove this case directly and we leads to the following lemma:

Lemma 1. *If $\mathcal{D} : e \hookrightarrow v$ and $\mathcal{C} : e \overset{v}{\leadsto} v'$ then $\mathcal{C}' : v \overset{v}{\leadsto} v'$.*

Proof. By Induction on the structure of $\mathcal{D}$.

In the following, we consider a few characteristic cases of the completeness proof in detail.

Proof.
The proof of the completeness theorem follows by induction on the structure of $\mathcal{D}$

Case $\mathcal{D} = e \hookrightarrow v$

$\mathcal{C} : e \overset{e}{\leadsto} \mathsf{vl}\ v'$	by assumption
$\mathcal{C} : e \overset{v}{\leadsto} v'$	by inversion on $\mathsf{c\_vl}$
$\mathcal{C}' : v \overset{v}{\leadsto} v'$	by lemma
$\mathcal{E} : \mathsf{vl}\ v' \overset{e}{\leadsto} v'$	by $\mathsf{ev\_vl}$

Case $\mathcal{D} = \cfrac{}{\mathbf{z} \hookrightarrow \mathbf{z}}\ \mathsf{ev\_z}$

By assumption we know $\mathcal{C} : \mathbf{z} \overset{e}{\leadsto} \mathsf{z}$. Then by rule $\mathsf{ev\_z}_v$ we know $\mathcal{E} : \mathsf{z} \overset{v}{\hookrightarrow} \mathsf{z}^*$ and by rule $\mathsf{cv\_z}$ we know $\mathcal{C}' : \mathbf{z} \overset{v}{\leadsto} \mathsf{z}^*$.

Case $\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1 \\ e \hookrightarrow v\end{array}}{\mathbf{s}\ e' \hookrightarrow v}\ \mathsf{ev\_s}$

$\mathcal{C} : \mathbf{s}\ e \overset{e}{\rightsquigarrow} \mathsf{s}\ e'$	by assumption
$\mathcal{C}_1 : e \overset{e}{\rightsquigarrow} e'$	by inversion on $\mathsf{c\_s}$
$\mathcal{E}_1 : e' \overset{v}{\hookrightarrow} v'$	by IH on $\mathcal{D}_1$
$\mathcal{C}_1' : v \overset{v}{\rightsquigarrow} v'$	
$\mathcal{E} : \mathsf{s}\ e' \overset{v}{\hookrightarrow} \mathsf{s}^*\ v'$	by $\mathsf{ev\_s}_v$
$\mathcal{C}' : \mathbf{s}\ v \overset{v}{\rightsquigarrow} \mathsf{s}^*\ v'$	by $\mathsf{cv\_s}$

Case $\mathcal{D} = \dfrac{}{\mathbf{lam}\ x.e \hookrightarrow \mathbf{lam}\ x.e}\ \mathsf{ev\_lam}$

$\mathcal{C} : \mathbf{lam}\ x.e \overset{e}{\rightsquigarrow} \mathsf{lam}\ y.e'$	by assumption
$\mathcal{C}_1 : u : x \overset{v}{\rightsquigarrow} y \vdash e \overset{e}{\rightsquigarrow} e'$	by inversion on $\mathsf{c\_lam}$
$\mathcal{C}' : \mathbf{lam}\ x.e \overset{v}{\rightsquigarrow} \mathsf{lam}^*\ y.e'$	by rule $\mathsf{cv\_lam}$ on $\mathcal{C}_1$
$\mathcal{E} : \mathsf{lam}\ y.e' \overset{v}{\hookrightarrow} \mathsf{lam}^*\ y.'$	by rule $\mathsf{ev\_lam}_v$

Case $\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1 \\ e_1 \hookrightarrow \mathbf{lam}\ x.e\end{array} \qquad \begin{array}{c}\mathcal{D}_2 \\ e_2 \hookrightarrow v_2\end{array} \qquad \begin{array}{c}\mathcal{D}_3 \\ [v_2/x]e \hookrightarrow v\end{array}}{(e_1\ e_2) \hookrightarrow v}\ \mathsf{ev\_app}$

$\mathcal{C} : e_1\ e_2 \overset{e}{\rightsquigarrow} e_1'\ e_2'$	by assumption
$\mathcal{C}_1 : e_1 \overset{e}{\rightsquigarrow} {e_1}'$	by inversion on $\mathcal{C}$ using $\mathsf{c\_app}$
$\mathcal{C}_2 : e_2 \overset{e}{\rightsquigarrow} {e_2}'$	
$\mathcal{E}_1 : e_1' \overset{v}{\hookrightarrow} v_1'$	by IH on $\mathcal{D}_1$, $\mathcal{C}_1$
$\mathcal{C}_1' : \mathbf{lam}\ x.e \overset{e}{\rightsquigarrow} v_1$	
$\mathcal{C}_3' : u : x \overset{v}{\rightsquigarrow} y \vdash e \overset{e}{\rightsquigarrow} e'$ and	by inversion on $\mathcal{C}_1'$ using $\mathsf{c\_lam}$
$v_1 = \mathsf{lam}\ y.e'$	
$\mathcal{E}_2 : e_2' \overset{v}{\hookrightarrow} v_2'$	by IH on $\mathcal{D}_2$, $\mathcal{C}_2$
$C_2' : v_2 \overset{v}{\rightsquigarrow} v_2'$	
$C_4' : [v_2/x]e \overset{e}{\rightsquigarrow} [v_2'/y]e'$	by substitution lemma $\mathcal{C}_3'$, $\mathcal{C}_2'$
$\mathcal{E}_3 : [v_2'/y]e' \overset{v}{\hookrightarrow} v'$	by IH on $\mathcal{D}_3$, $\mathcal{C}_4'$
$C' : v \overset{v}{\rightsquigarrow} v'$	
$\mathcal{E} : e_1'e_2' \overset{v}{\hookrightarrow} v'$	by $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ using $\mathsf{ev\_app}_v$

Soundness Proof

Theorem 4. *If* $\mathcal{E} : e' \overset{v}{\hookrightarrow} v'$ *and* $\mathcal{C} : e \overset{e}{\rightsquigarrow} e'$ *then* $\mathcal{D} : e \hookrightarrow v$ *and* $\mathcal{C}' : v \overset{v}{\rightsquigarrow} v'$.

Proof. The proof is by induction on the structure of $\mathcal{E}$. The proof follows the same principles as the completeness proof.

Implementation

The implementation in Twelf covers the full mini-ml language and all the cases for the completeness and soundness proofs.

Solution without separating values and expressions

Of course there are many solutions to the problem stated in exercise 2.7. To highlight some of the advantages of the presented solution, we briefly outline a solution which does not explicitly distinguish between values and expressions and compare it to the solution presented earlier. Any expression which has already been evaluated and is substituted into another expression is marked with vl .

$$\frac{}{\mathsf{z} \overset{v}{\hookrightarrow} \mathsf{z}}\ \mathsf{ev\_z}_v \qquad \frac{e' \overset{v}{\hookrightarrow} v'}{\mathsf{s}\ e' \overset{v}{\hookrightarrow} (\mathsf{s}\ v')}\ \mathsf{ev\_s}_v$$

$$\frac{e_1 \overset{v}{\hookrightarrow} \mathsf{z} \qquad e_2 \overset{v}{\hookrightarrow} v}{\mathsf{case}\ e_1\ \mathsf{of}\ \mathsf{z}^* \Rightarrow e_2 \,|\, \mathsf{s}^*\ x \Rightarrow e_3 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_case\_z}_v \qquad \frac{e_1 \overset{v}{\hookrightarrow} (\mathsf{s}\ v_2) \qquad [\mathsf{vl}\ v_2/x]e_3 \overset{v}{\hookrightarrow} v}{\mathsf{case}\ e_1\ \mathsf{of}\ \mathsf{z}^* \Rightarrow e_2 \,|\, \mathsf{s}^*\ x \Rightarrow e_3 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_case\_s}_v$$

$$\frac{}{\mathsf{lam}\ x.e \overset{v}{\hookrightarrow} \mathsf{lam}\ x.e}\ \mathsf{ev\_lam}_v$$

$$\frac{e_1 \overset{v}{\hookrightarrow} \mathsf{lam}\ x.e' \qquad e_2 \overset{v}{\hookrightarrow} v_2 \qquad [\mathsf{vl}\ v_2/x]e' \overset{v}{\hookrightarrow} v}{e_1 e_2 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_app}_v$$

$$\frac{e_1 \overset{v}{\hookrightarrow} v_1 \qquad [\mathsf{vl}\ v_1/x]e_2 \overset{v}{\hookrightarrow} v}{\mathsf{let\ val}\ x = e_1\ \mathsf{in}\ e_2 \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_letv}_v$$

$$\frac{}{\mathsf{vl}\ v \overset{v}{\hookrightarrow} v}\ \mathsf{ev\_vl}$$

We only show the conversion rule for functions.

$$\cfrac{\begin{array}{c}\cfrac{}{x \overset{e}{\rightsquigarrow} y}\,u \\ \vdots \\ e \overset{e}{\rightsquigarrow} e'\end{array}}{\textbf{lam}\ x.e \overset{e}{\rightsquigarrow} \mathsf{lam}\ y.e'}\ \mathsf{c\_lam}^{x,y,u}$$

We will discuss the completeness proof next.

Theorem 5 (Completeness). *If $\mathcal{D} : e \hookrightarrow v$ and $\mathcal{C} : e \overset{e}{\rightsquigarrow} e'$ then there exists an evaluation $\mathcal{E} : e' \overset{v}{\hookrightarrow} v'$ and $\mathcal{C}' : v \overset{e}{\rightsquigarrow} v'$.*

Proof. Proof by induction on the structure of $\mathcal{D}$. We will only consider the case for ev_app rule.

$$\textbf{Case}\ \mathcal{D} = \cfrac{\begin{array}{c}\mathcal{D}_1 \\ e_1 \hookrightarrow \textbf{lam}\ x.e\end{array} \qquad \begin{array}{c}\mathcal{D}_2 \\ e_2 \hookrightarrow v_2\end{array} \qquad \begin{array}{c}\mathcal{D}_3 \\ [v_2/x]e \hookrightarrow v\end{array}}{(e_1\ e_2) \hookrightarrow v}\ \mathsf{ev\_app}$$

$\mathcal{C} : e_1\ e_2 \overset{e}{\rightsquigarrow} e_1'\ e_2'$	by assumption
$\mathcal{C}_1 : e_1 \overset{e}{\rightsquigarrow} {e_1}'$	by inversion on $\mathcal{C}$ using c_app
$\mathcal{C}_2 : e_2 \overset{e}{\rightsquigarrow} {e_2}'$	
$\mathcal{E}_1 : e_1' \overset{v}{\hookrightarrow} v_1'$	by IH on $\mathcal{D}_1$, $\mathcal{C}_1$
$\mathcal{C}_1' : \textbf{lam}\ x.e \overset{e}{\rightsquigarrow} v_1'$	

We would like to apply inversion on $\mathcal{C}_1'$ to determine the value v_1. However, two inference rules could have been applied: c_lam and c_vl. In fact there is a hidden invariant that v_1 must be a value where values are defined as before except that **lam** $x.e$ now may contain vl . To appeal to the inversion principle, we need to prove first a value soundness theorem about the new semantics. Therefore we know that v_1 must be a value and the only possible rule which could have been applied to $\mathcal{C}_1'$ where v_1' is a value is the c_lam rule.

$v_1 = \mathsf{lam}\ y.e'$	
$\mathcal{C}_3' : u : x \overset{e}{\rightsquigarrow} y \vdash e \overset{e}{\rightsquigarrow} [\mathsf{vl}\ y/y]e'$ and	by inversion on $\mathcal{C}_1'$ using c_lam
$\mathcal{E}_2 : e_2' \overset{v}{\hookrightarrow} v_2'$	by IH on $\mathcal{D}_2$, $\mathcal{C}_2$
$C_2' : v_2 \overset{e}{\rightsquigarrow} v_2'$	
$C_4' : [v_2/x]e \overset{e}{\rightsquigarrow} [\mathsf{vl}\ v_2'/y]e'$	by substitution lemma $\mathcal{C}_3'$, $\mathcal{C}_2'$
$\mathcal{E}_3 : [v_2'/y]e' \overset{v}{\hookrightarrow} v'$	by IH on $\mathcal{D}_3$, $\mathcal{C}_4'$
$C' : v \overset{e}{\rightsquigarrow} v'$	
$\mathcal{E} : e_1'e_2' \overset{v}{\hookrightarrow} v'$	by $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ using $\mathsf{ev\_app}_v$

Case $\mathcal{D} = e \hookrightarrow v$

$\mathcal{C} : e \overset{e}{\rightsquigarrow} \mathsf{vl}\ e'$ by assumption

To prove this case, we need to first show that e and e' are values. Second we prove that values evaluate to themselves in both semantics. Therefore there exists a $\mathcal{E} : e' \overset{v}{\hookrightarrow} e'$. By assumption we know $\mathcal{C} : e \overset{e}{\rightsquigarrow} (vle')$. By inversion on $\mathcal{C}$, we obtain $\mathcal{C}' : e \overset{e}{\rightsquigarrow} e'$ which is what we needed to prove.

There are a lot of choices to be made when designing a new semantics. The distinction between values and expressions in the first solution provides guidance through the design and development of the semantics and proofs about them. Implementing the semantics and proofs is relatively straightforward if structural properties are exposed. In addition, type checking will enforce invariants such as the value soundness property.

If we do not distinguish between expressions and values, we must prove properties like value soundness separately. In addition, the appeal to the inversion principle is not straightforward and we need to prove additional lemmata to handle the c_vl case. This causes considerable overhead. Finally, it is easy to make mistakes if we do not distinguish between values and expressions.