Mini-ML operational semantics (from slides from JC Filliâtre)

Laure Gonnord

October 10, 2018

Abstract syntax

Recall the abstract syntax of miniML (without recursion):

e	$::=$	x	identificator
	$\mid$	c	constant (1, 2, ..., *true*, ...)
	$\mid$	op	primitive ($+$, $\times$, *fst*, ...)
	$\mid$	$\texttt{fun}\ x \to e$	function
	$\mid$	$e\ e$	application
	$\mid$	(e, e)	pair
	$\mid$	$\texttt{let}\ x = e\ \texttt{in}\ e$	local binding

1 Semantic rules

1.1 Natural "big steps" semantics (NAT)

We define the relation:

$$e \xrightarrow{v} v$$

where values (v) have the following abstract syntax:

v	$::=$	c	constant
	$\mid$	op	primitive
	$\mid$	$\texttt{fun}\ x \to e$	function
	$\mid$	(v, v)	pair

Name	Rule
cste	$c \xrightarrow{v} c$
op	$op \xrightarrow{v} op$
fun	$(\texttt{fun}\ x \to e) \xrightarrow{v} (\texttt{fun}\ x \to e)$
locvar	$\dfrac{e_1 \xrightarrow{v} v_1 \quad e_2[x \leftarrow v_1] \xrightarrow{v} v}{\texttt{let}\ x{=}e_1\ \texttt{in}\ e_2 \xrightarrow{v} v}$
apply	$\dfrac{e_1 \xrightarrow{v} (\texttt{fun}\ x \to e) \quad e_2 \xrightarrow{v} v_2 \quad e[x \leftarrow v_2] \xrightarrow{v} v}{e_1\ e_2 \xrightarrow{v} v}$
primitives	$\dfrac{e_1 \xrightarrow{v} + \quad e_2 \xrightarrow{v} (n_1, n_2) \quad n = n_1 + n_2}{e_1\ e_2 \xrightarrow{v} n}$
tuples	$\dfrac{e_1 \xrightarrow{v} \mathit{fst} \quad e_2 \xrightarrow{v} (v_1, v_2)}{e_1\ e_2 \xrightarrow{v} v_1}$

Proposition 1 (closed terms)**.** *If $e \stackrel{v}{\to} v$ then v is a value. Moreover, if e is closed (no free variable) then v is also closed.*

Proposition 2. *Determinism If $e \stackrel{v}{\to} v$ and $e \stackrel{v}{\to} v'$ then $v = v'$.*

1.2 Reduction semantics (small steps)

$$e \to e_1 \to e_2 \to \cdots$$

Then iterations may: finish with a value, or block on an irreducible expression, or do not terminate.

We first define a "head reduction" $\stackrel{\epsilon}{\to}$ at the toplevel of an expression:

$$(\texttt{fun}\ x \to e)\ v \quad \stackrel{\epsilon}{\to} \quad e[x \leftarrow v]$$

$$\texttt{let}\ x = v\ \texttt{in}\ e \quad \stackrel{\epsilon}{\to} \quad e[x \leftarrow v]$$

Then rules for primitives:

$$+\ (n_1, n_2) \quad \stackrel{\epsilon}{\to} \quad n \qquad \text{avec } n = n_1 + n_2$$

$$\begin{aligned} \mathit{fst}\ (v_1, v_2) &\quad \stackrel{\epsilon}{\to} \quad v_1 \\ \mathit{snd}\ (v_1, v_2) &\quad \stackrel{\epsilon}{\to} \quad v_2 \end{aligned}$$

Now we have to introduce "deep reduction" (in order to evaluate subexpressions):

$$\frac{e_1 \stackrel{\epsilon}{\to} e_2}{E(e_1) \to E(e_2)}$$

where E is a context, with the following syntax:

$$\begin{aligned} E \quad ::= \quad & \Box \\ | \quad & E\ e \\ | \quad & v\ E \\ | \quad & \texttt{let}\ x = E\ \texttt{in}\ e \\ | \quad & (E, e) \\ | \quad & (v, E) \end{aligned}$$

Example 1.

$$E \stackrel{def}{=} \texttt{let}\ x = +(2, \Box)\ \texttt{in}\ \texttt{let}\ y = +(x, x)\ \texttt{in}\ y$$

Definition 1 (substitution)**.** *$E(e)$ denotes the context E where $\Box$ has been replaced by e.*

Example 2.

$$E(+(10, 9)) = \texttt{let}\ x = +(2, +(10, 9))\ \texttt{in}\ \texttt{let}\ y = +(x, x)\ \texttt{in}\ y$$

A context is a "term with a hole", where $\Box$ is the hole.

Example 3.

$$\frac{+(1,2) \stackrel{\epsilon}{\rightarrow} 3}{\texttt{let } x = +(1,2) \texttt{ in } +(x,x) \rightarrow \texttt{let } x = 3 \texttt{ in } +(x,x)}$$

thanks to the context $E \stackrel{def}{=} \texttt{let } x = \Box \texttt{ in } +(x,x)$

Definition 2. *We denote by* $\stackrel{\star}{\rightarrow}$ *the reflexive and transitive closure of* $\rightarrow$*.*

Definition 3. *A normal form is any "unreductible expression", ie an expression for which there is no* e' *such that* $e \rightarrow e'$*.*

2 Equivalence

Theorem 1.

$$e \stackrel{v}{\rightarrow} v \quad \textit{if and only if} \quad e \stackrel{\star}{\rightarrow} v$$

Big steps implies small step The demo is based on the following lemma:

Lemma 1. *Let us suppose* $e \rightarrow e'$*. Then for every expression* e_2 *and value* v*:*

1. $e\ e_2 \rightarrow e'\ e_2$
2. $v\ e \rightarrow v\ e'$
3. $\texttt{let } x = e \texttt{ in } e_2 \rightarrow \texttt{let } x = e' \texttt{ in } e_2$

Proposition 3. *If* $e \stackrel{v}{\rightarrow} v$*, then* $e \stackrel{\star}{\rightarrow} v$*.*

Small steps to big steps First, some lemmas:

Lemma 2. $v \stackrel{v}{\rightarrow} v$ *for every value* v*.*

Lemma 3 (reduction and evaluation)**.** *If* $e \rightarrow e'$ *and* $e' \stackrel{v}{\rightarrow} v$*, then* $e \stackrel{v}{\rightarrow} v$*.*

Lemma 4. *If* $e \stackrel{\epsilon}{\rightarrow} e'$ *and* $E(e') \stackrel{v}{\rightarrow} v$ *then* $E(e) \stackrel{v}{\rightarrow} v$*.*

Proposition 4 (small steps to big steps)**.** *If* $e \stackrel{\star}{\rightarrow} v$*, then* $e \stackrel{v}{\rightarrow} v$*.*