GhostML : a mini-ML with global references and ghost terms

prog	::=	*typedecl** *vardecl** *t*	program
typedecl	::=	`type` *id* $\alpha, ..., \alpha$ $= \tau$	type declaration
vardecl	::=	`val` *id* : `ref` τ	global reference declaration

GhostML Programs

τ	::=	α	type variable
	|	$\varepsilon\ (\tau, \ldots, \tau)$	datatype constructor
	|	$\tau \to \tau$	function type
	|	`int` | `bool` | `Prop` | ...	build-in types
σ	::=	$\forall\overline{\alpha}.\tau$	type scheme

GhostML Types and Schemes

v	::=	x	variable
	|	op	build-in constants and operands (*1*, *true*, +, $\vee$, ...)
	|	$C(v, \ldots, v)$	constructor application
	|	$(\texttt{fun}\ \varepsilon x \to t)$	function

GhostML Values

t	::=	v	value
	|	$v(v)$	application
	|	`let` $x = t$ `in` t	local binding
	|	`letrec` $f\ x = t$	recursive function
	|	$!\ x$	global reference access
	|	$x := t$	global reference assignment
	|	if t then t else t	conditional expression
	|	match t with $p \to t,\ \ldots,\ p \to t$ end	pattern-matching

GhostML Terms

p	::=	x	variable pattern
	|	$C(p, \ldots, p)$	constructor pattern

GhostML Patterns

Operational Semantics

$$(\textbf{fun}\ \varepsilon x_\tau \to t) v_{|\sigma} \quad \rightsquigarrow \quad t[x \longleftarrow v]_{|\sigma} \qquad \text{(E-App-Redex)}$$

$$\textbf{let}\ \varepsilon x = v_1\ \textbf{in}\ t_{|\sigma} \quad \rightsquigarrow \quad t[x \longleftarrow v]_{|\sigma} \qquad \text{(E-Let-Redex)}$$

$$\textbf{if}\ \texttt{true}\ \textbf{then}\ t_1\ \textbf{else}\ t_{2|\sigma} \quad \rightsquigarrow \quad t_{1|\sigma} \qquad \text{(E-IF-True-Redex)}$$

$$\textbf{if}\ \texttt{false}\ \textbf{then}\ t_1\ \textbf{else}\ t_{2|\sigma} \quad \rightsquigarrow \quad t_{2|\sigma} \qquad \text{(E-IF-False-Redex)}$$

$$\frac{t_{1|\sigma} \quad \rightsquigarrow \quad t'_{1|\sigma'}}{\textbf{if}\ t_1\ \textbf{then}\ t_2\ \textbf{else}\ t_{3|\sigma} \quad \rightsquigarrow \quad \textbf{if}\ t'_1\ \textbf{then}\ t_2\ \textbf{else}\ t_{3|\sigma'}} \qquad \text{(E-IF-Context)}$$

$$\frac{t_{1|\sigma} \quad \rightsquigarrow \quad t'_{1|\sigma'}}{\textbf{let}\ \varepsilon x = t_1\ \textbf{in}\ t_{2|\sigma} \quad \rightsquigarrow \quad \textbf{let}\ \varepsilon x = t_1\ \textbf{in}\ t'_{2|\sigma'}} \qquad \text{(E-Let-Context)}$$