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

$prog$	$::=$	$typedecl^{*} \quad vardecl^{*} \quad t$	program
$typedecl$	$::=$	$\texttt{type}\ id\ \alpha, ..., \alpha\ = \tau$	type declaration
$vardecl$	$::=$	$\texttt{val}\ id : \texttt{ref}\ \tau$	global reference declaration

GhostML Programs

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

GhostML Types and Schemes

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

GhostML Values

t	$::=$	v	value
	$\mid$	$t(t)$	application
	$\mid$	$C(t, \ldots, t)$	constructor application
	$\mid$	$\texttt{let} x = t\ \texttt{in}\ t$	local binding
	$\mid$	$\texttt{letrec} f x = t$	recursive function
	$\mid$	$\texttt{ghost}\ t$	ghost term
	$\mid$	$!\ x$	global reference access
	$\mid$	$x := t$	global reference assignment
	$\mid$	$\mathsf{if}\ t\ \mathsf{then}\ t\ \mathsf{else}\ t$	conditional expression
	$\mid$	$\mathsf{match}\ t\ \mathsf{with}\ p \rightarrow t,\ \ldots,\ p \rightarrow t\ \mathsf{end}$	pattern-matching

GhostML Terms

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

GhostML Patterns