

<i>ident, x, y, y<sub>p</sub>, y<sub>f</sub>, -, abbrev, r</i>	subscripts: p for pointers, f for functions
<i>n, i, j</i>	index variables
<i>impl_const</i>	implementation-defined constant
<i>member</i>	C struct/union member name
	Ott-hack, ignore (annotations)
<i>nat</i>	OCaml arbitrary-width natural number
<i>mem_ptr</i>	abstract pointer value
<i>mem_val</i>	abstract memory value
	Ott-hack, ignore (locations)
<i>mem_iv_c</i>	OCaml type for memory constraints on integer values
<i>UB_name</i>	undefined behaviour
<i>string</i>	OCaml string
	Ott-hack, ignore (OCaml type variable TY)
	Ott-hack, ignore (OCaml Symbol.prefix)
<i>mem_order, _</i>	OCaml type for memory order
<i>linux_mem_order</i>	OCaml type for Linux memory order
	Ott-hack, ignore (OCaml type variable bt)

$Stypes\_t, \tau$	$::=$	C type
	$\tau^*$	pointer to type $\tau$
$tag$	$::=$	OCaml type for struct/union tag
	$ident$	
$\beta, -$	$::=$	base types
	<b>unit</b>	unit
	<b>bool</b>	boolean
	<b>integer</b>	integer
	<b>real</b>	rational numbers?
	<b>loc</b>	location
	<b>array</b> $\beta$	array
	<b>list</b> $\beta$	list
	$\overline{\beta_i}^i$	tuple
	<b>struct</b> $tag$	struct
	<b>set</b> $\beta$	set
	<b>opt</b> $(\beta)$	option
	$\beta \rightarrow \beta'$	parameter types
	$\beta_\tau$ M	of a C type
$binop$	$::=$	binary operators
	<b>+</b>	addition
	<b>-</b>	subtraction
	<b>*</b>	multiplication
	<b>/</b>	division
	<b>rem_t</b>	modulus
	<b>rem_f</b>	remainder
	<b>^</b>	exponentiation
	<b>=</b>	equality, defined both for integer and C types

		!=	inequality, similiarly defined
		>	greater than, similarly defined
		<	less than, similarly defined
		>=	greater than or equal to, similarly defined
		<=	less than or equal to, similarly defined
		/\	conjuction
		\/	disjunction
<i>binop<sub>arith</sub></i>	::=		arithmetic binary operators
		+	
		-	
		*	
		/	
		rem_t	
		rem_f	
		^	
<i>binop<sub>rel</sub></i>	::=		relational binary operators
		=	
		!=	
		>	
		<	
		>=	
		<=	
<i>binop<sub>bool</sub></i>	::=		boolean binary operators
		/\	
		\/	
<i>mem_int</i>	::=		memory integer value

	1	M
	0	M
<i>object_value</i>	$::=$   <i>mem_int</i>   <i>mem_ptr</i>   $\text{array}(\overline{\text{loaded\_value}_i}^i)$   $(\text{struct } \textit{ident})\{\overline{\text{member}_i:\tau_i = \text{mem\_val}_i}^i\}$   $(\text{union } \textit{ident})\{\text{.member} = \text{mem\_val}\}$	C object values (inhabitants of object types), which can be read/stored integer value pointer value C array value C struct value C union value
<i>loaded_value</i>	$::=$   <i>specified object_value</i>	potentially unspecified C object values specified loaded value
<i>value</i>	$::=$   <i>object_value</i>   <i>loaded_value</i>   <b>Unit</b>   <b>True</b>   <b>False</b>   $\beta[\overline{\text{value}_i}^i]$   $(\overline{\text{value}_i}^i)$	Core values C object value loaded C object value unit boolean true boolean false list tuple
<i>bool_value</i>	$::=$   <b>True</b>   <b>False</b>	Core booleans boolean true boolean false
<i>ctor_val</i>	$::=$   <b>Nil</b> $\beta$   <b>Cons</b>   <b>Tuple</b>	data constructors empty list list cons tuple

		Array	C array
		Specified	non-unspecified loaded value
<i>ctor_expr</i>	::=		data constructors
		Ivmax	max integer value
		Ivmin	min integer value
		Ivsizeof	sizeof value
		Ivalignof	alignof value
		IvCOMPL	bitwise complement
		IvAND	bitwise AND
		IvOR	bitwise OR
		IvXOR	bitwise XOR
		Fvfromint	cast integer to floating value
		Ivfromfloat	cast floating to integer value
<i>name</i>	::=		
		<i>ident</i>	Core identifier
		<i>impl_const</i>	implementation-defined constant
<i>pval</i>	::=		pure values
		<i>ident</i>	Core identifier
		<i>impl_const</i>	implementation-defined constant
		<i>value</i>	Core values
		$\text{constrained}(\overline{\text{mem\_iv\_c}_i, \text{pval}_i}^i)$	constrained value
		$\text{error}(\text{string}, \text{pval})$	impl-defined static error
		$\text{ctor\_val}(\overline{\text{pval}_i}^i)$	data constructor application
		$(\text{struct } \text{ident})\{\overline{\text{member}_i = \text{pval}_i}^i\}$	C struct expression
		$(\text{union } \text{ident})\{\text{member} = \text{pval}\}$	C union expression
<i>tpval</i>	::=		top-level pure values

		<b>undef</b> <i>UB_name</i>		undefined behaviour
		<b>done</b> <i>pval</i>		pure done
<i>ident_opt_β</i>	::=			type annotated optional identifier
		<i>_.β</i>	binders = {}	
		<i>ident:β</i>	binders = <i>ident</i>	
<i>pattern</i>	::=			
		<i>ident_opt_β</i>	binders = binders( <i>ident_opt_β</i> )	
		<i>ctor_val(⟦pattern<sub>i</sub>⟧<sup>i</sup>)</i>	binders = binders( <i>pattern<sub>i</sub></i> )	
<i>z</i>	::=			OCaml arbitrary-width integer
		<i>i</i>	M	literal integer
		<i>mem_int</i>	M	
		<i>size_of(τ)</i>	M	size of a C type
		<i>offset_of<sub>tag</sub>(member)</i>	M	offset of a struct member
		<b>ptr_size</b>	M	size of a pointer
		<i>max_int<sub>τ</sub></i>	M	maximum value of int of type τ
		<i>min_int<sub>τ</sub></i>	M	minimum value of int of type τ
$\mathbb{Q}$ , <i>q</i> , -	::=			OCaml type for rational numbers
		$\frac{int_1}{int_2}$		
<i>lit</i>	::=			
		<i>ident</i>		
		<b>unit</b>		
		<i>bool</i>		
		<i>z</i>		
		$\mathbb{Q}$		

<i>ident_or_pattern</i>	$::=$ $\begin{array}{l}   \textit{ident} \\   \textit{pattern} \end{array}$	$\text{binders} = \textit{ident}$ $\text{binders} = \text{binders}(\textit{pattern})$
<i>bool_op</i>	$::=$ $\begin{array}{l}   \neg \textit{term} \\   \textit{term}_1 = \textit{term}_2 \\   \bigwedge (\overline{\textit{term}_i}^i) \\   \bigvee (\overline{\textit{term}_i}^i) \\   \textit{term}_1 \textit{binop}_{\textit{bool}} \textit{term}_2 \\   \text{if } \textit{term}_1 \text{ then } \textit{term}_2 \text{ else } \textit{term}_3 \end{array}$	$\text{M}$
<i>arith_op</i>	$::=$ $\begin{array}{l}   \textit{term}_1 + \textit{term}_2 \\   \textit{term}_1 - \textit{term}_2 \\   \textit{term}_1 \times \textit{term}_2 \\   \textit{term}_1 / \textit{term}_2 \\   \textit{term}_1 \text{rem\_t } \textit{term}_2 \\   \textit{term}_1 \text{rem\_f } \textit{term}_2 \\   \textit{term}_1 \wedge \textit{term}_2 \\   \textit{term}_1 \textit{binop}_{\textit{arith}} \textit{term}_2 \end{array}$	$\text{M}$
<i>cmp_op</i>	$::=$ $\begin{array}{l}   \textit{term}_1 < \textit{term}_2 \\   \textit{term}_1 \leq \textit{term}_2 \\   \textit{term}_1 \textit{binop}_{\textit{rel}} \textit{term}_2 \end{array}$	$\begin{array}{l} \text{less than} \\ \text{less than or equal} \end{array}$ $\text{M}$
<i>list_op</i>	$::=$ $\begin{array}{l}   \text{nil} \\   \textit{term}_1 :: \textit{term}_2 \end{array}$	

		<b>tl</b> $term$
		$term^{(int)}$
$tuple\_op$	$::=$	
		$(\overline{term_i})^i$
		$term^{(int)}$
$pointer\_op$	$::=$	
		$mem\_ptr$
		$term_1 +_{ptr} term_2$
		<b>cast_int_to_ptr</b> $term$
		<b>cast_ptr_to_int</b> $term$
$array\_op$	$::=$	
		$[[ \overline{term_i}^i ]]$
		$term_1[term_2]$
$param\_op$	$::=$	
		$ident:\beta. term$
		$term(term_1, \dots, term_n)$
$struct\_op$	$::=$	
		$term.member$
$ct\_pred$	$::=$	
		<b>representable</b> $(\tau, term)$
		<b>aligned</b> $(\tau, term)$
		<b>alignedI</b> $(term_1, term_2)$
$term, \_$	$::=$	



		<i>lit</i>	
		<i>arith_op</i>	
		<i>bool_op</i>	
		<i>cmp_op</i>	
		<i>tuple_op</i>	
		<i>struct_op</i>	
		<i>pointer_op</i>	
		<i>list_op</i>	
		<i>array_op</i>	
		<i>ct_pred</i>	
		<i>param_op</i>	
		$(term)$	S    parentheses
		$\sigma(term)$	M    simul-sub $\sigma$ in <i>term</i>
		<i>pval</i>	M
<i>pexpr</i>	<b>::=</b>		pure expressions
		<i>pval</i>	pure values
		<i>ctor_expr</i> ( $\overline{pval_i}^i$ )	data constructor application
		<b>array_shift</b> ( <i>pval</i> <sub>1</sub> , $\tau$ , <i>pval</i> <sub>2</sub> )	pointer array shift
		<b>member_shift</b> ( <i>pval</i> , <i>ident</i> , <i>member</i> )	pointer struct/union member shift
		<b>not</b> ( <i>pval</i> )	boolean not
		<i>pval</i> <sub>1</sub> <i>binop</i> <i>pval</i> <sub>2</sub>	binary operations
		<b>memberof</b> ( <i>ident</i> , <i>member</i> , <i>pval</i> )	C struct/union member access
		<i>name</i> ( $\overline{pval_i}^i$ )	pure function call
		<b>assert_undef</b> ( <i>pval</i> , <i>UB_name</i> )	
		<b>bool_to_integer</b> ( <i>pval</i> )	
		<b>conv_int</b> ( $\tau$ , <i>pval</i> )	
		<b>wrapI</b> ( $\tau$ , <i>pval</i> )	
<i>tpexpr</i>	<b>::=</b>		top-level pure expressions

	$tpval$		top-level pure values
	<b>case</b> $pval$ <b>of</b> $\overline{texpr\_case\_branch_i}^i$ <b>end</b>		pattern matching
	<b>let</b> $ident\_or\_pattern = pexpr$ <b>in</b> $texpr$	bind $binders(ident\_or\_pattern)$ in $texpr$	pure let
	<b>let</b> $ident\_or\_pattern:(y_1:\beta_1. term_1) = texpr_1$ <b>in</b> $texpr_2$	bind $binders(ident\_or\_pattern)$ in $texpr_2$	pure let
	<b>if</b> $pval$ <b>then</b> $texpr_1$ <b>else</b> $texpr_2$	bind $y_1$ in $term_1$	pure if
	$\sigma(texpr)$	M	simul-sub $\sigma$ in $texpr$
$texpr\_case\_branch$	::=		pure top-level case expression
	$pattern \Rightarrow texpr$	bind $binders(pattern)$ in $texpr$	top-level case expression br
$m\_kill\_kind$	::=		
	<b>dynamic</b>		
	<b>static</b> $\tau$		
$bool, -$	::=		OCaml booleans
	<b>true</b>		
	<b>false</b>		
$int, -$	::=		OCaml fixed-width integer
	$i$		literal integer
$res\_term$	::=		resource terms
	<b>emp</b>		empty heap
	$points\_to$		single-cell heap
	$ident$		variable
	$\langle res\_term_1, res\_term_2 \rangle$		seperating-conjunction pair
	<b>pack</b> $(pval, res\_term)$		packing for existentials
	$\sigma(res\_term)$	M	substitution for resource te

<i>mem_action</i>	$::=$ <ul style="list-style-type: none"> <li>  <code>create</code> (<i>pval</i>, <math>\tau</math>)</li> <li>  <code>create_readonly</code> (<i>pval</i><sub>1</sub>, <math>\tau</math>, <i>pval</i><sub>2</sub>)</li> <li>  <code>alloc</code> (<i>pval</i><sub>1</sub>, <i>pval</i><sub>2</sub>)</li> <li>  <code>kill</code> (<i>m_kill_kind</i>, <i>pval</i>, <i>pt</i>)</li> <li>  <code>store</code> (<i>bool</i>, <math>\tau</math>, <i>pval</i><sub>1</sub>, <i>pval</i><sub>2</sub>, <i>mem_order</i>, <i>pt</i>)</li> <li>  <code>load</code> (<math>\tau</math>, <i>pval</i>, <i>mem_order</i>, <i>pt</i>)</li> <li>  <code>rmw</code> (<math>\tau</math>, <i>pval</i><sub>1</sub>, <i>pval</i><sub>2</sub>, <i>pval</i><sub>3</sub>, <i>mem_order</i><sub>1</sub>, <i>mem_order</i><sub>2</sub>)</li> <li>  <code>fence</code> (<i>mem_order</i>)</li> <li>  <code>cmp_exch_strong</code> (<math>\tau</math>, <i>pval</i><sub>1</sub>, <i>pval</i><sub>2</sub>, <i>pval</i><sub>3</sub>, <i>mem_order</i><sub>1</sub>, <i>mem_order</i><sub>2</sub>)</li> <li>  <code>cmp_exch_weak</code> (<math>\tau</math>, <i>pval</i><sub>1</sub>, <i>pval</i><sub>2</sub>, <i>pval</i><sub>3</sub>, <i>mem_order</i><sub>1</sub>, <i>mem_order</i><sub>2</sub>)</li> <li>  <code>linux_fence</code> (<i>linux_mem_order</i>)</li> <li>  <code>linux_load</code> (<math>\tau</math>, <i>pval</i>, <i>linux_mem_order</i>)</li> <li>  <code>linux_store</code> (<math>\tau</math>, <i>pval</i><sub>1</sub>, <i>pval</i><sub>2</sub>, <i>linux_mem_order</i>)</li> <li>  <code>linux_rmw</code> (<math>\tau</math>, <i>pval</i><sub>1</sub>, <i>pval</i><sub>2</sub>, <i>linux_mem_order</i>)</li> </ul>	<p>memory actions</p> <p>true means store is locking</p>
<i>polarity</i>	$::=$ <ul style="list-style-type: none"> <li> </li> <li>  <code>neg</code></li> </ul>	<p>polarities for memory actions</p> <p>(pos) sequenced by <code>let weak</code> and <code>let strong</code></p> <p>only sequenced by <code>let strong</code></p>
<i>pol_mem_action</i>	$::=$ <ul style="list-style-type: none"> <li>  <i>polarity mem_action</i></li> </ul>	<p>memory actions with polarity</p>
<i>mem_op</i>	$::=$ <ul style="list-style-type: none"> <li>  <i>pval</i><sub>1</sub> <i>binop<sub>rel</sub></i> <i>pval</i><sub>2</sub></li> <li>  <i>pval</i><sub>1</sub> <math>-_{\tau}</math> <i>pval</i><sub>2</sub></li> <li>  <code>intFromPtr</code> (<math>\tau</math><sub>1</sub>, <math>\tau</math><sub>2</sub>, <i>pval</i>)</li> <li>  <code>ptrFromInt</code> (<math>\tau</math><sub>1</sub>, <math>\tau</math><sub>2</sub>, <i>pval</i>)</li> <li>  <code>ptrValidForDeref</code> (<math>\tau</math>, <i>pval</i>, <i>pt</i>)</li> <li>  <code>ptrWellAligned</code> (<math>\tau</math>, <i>pval</i>)</li> </ul>	<p>operations involving the memory state</p> <p>pointer relational binary operations</p> <p>pointer subtraction</p> <p>cast of pointer value to integer value</p> <p>cast of integer value to pointer value</p> <p>dereferencing validity predicate</p>

		<code>ptrArrayShift</code> ( $pval_1, \tau, pval_2$ )		
		<code>memcpy</code> ( $pval_1, pval_2, pval_3$ )		
		<code>memcmp</code> ( $pval_1, pval_2, pval_3$ )		
		<code>realloc</code> ( $pval_1, pval_2, pval_3$ )		
		<code>va_start</code> ( $pval_1, pval_2$ )		
		<code>va_copy</code> ( $pval$ )		
		<code>va_arg</code> ( $pval, \tau$ )		
		<code>va_end</code> ( $pval$ )		
$spine\_elem$	$::=$			spine element
		$pval$		pure or logical value
		$res\_term$		resource value
		$\sigma(spine\_elem)$	M	substitution for spine elements / return values
$spine$	$::=$			spine
		$\overline{spine\_elem_i}^i$		
$tval$	$::=$			(effectful) top-level values
		<code>done</code> $spine$		end of top-level expression
		<code>undef</code> $UB\_name$		undefined behaviour
$res\_pattern$	$::=$			resource terms
		<code>emp</code>	binders = {}	empty heap
		$pt$	binders = {}	single-cell heap
		$ident$	binders = $ident$	variable
		$\langle res\_pattern_1, res\_pattern_2 \rangle$	binders = $binders(res\_pattern_1) \cup binders(res\_pattern_2)$	seperating-conjunction pair
		<code>pack</code> ( $ident, res\_pattern$ )	binders = $ident \cup binders(res\_pattern)$	packing for existentials
$ret\_pattern$	$::=$			return pattern
		<code>comp</code> $ident\_or\_pattern$	binders = $binders(ident\_or\_pattern)$	computational variable

		$\text{log } ident$	$\text{binders} = ident$	logical variable
		$\text{res } res\_pattern$	$\text{binders} = \text{binders}(res\_pattern)$	resource variable
$init,$	$::=$			initialisation status
		$\checkmark$		initialised
		$\times$		uninitialised
$points\_to, pt$	$::=$			points-to separation logic predicate
		$term_1 \xrightarrow{init}_{\tau} term_2$		
$res$	$::=$			resources
		$\text{emp}$		empty heap
		$points\_to$		points-top heap pred.
		$res_1 * res_2$		seperating conjunction
		$\exists ident:\beta. res$		existential
		$term \wedge res$		logical conjunction
		$\sigma(res)$	M	simul-sub $\sigma$ in $res$
$ret, -$	$::=$			return types
		$\Sigma ident:\beta. ret$		return a computational value
		$\exists ident:\beta. ret$		return a logical value
		$res \otimes ret$		return a resource value
		$term \wedge ret$		return a predicate (post-condition)
		$I$		end return list
		$\sigma(ret)$	M	simul-sub $\sigma$ in $ret$
$seq\_expr$	$::=$			sequential (effectful) expressions
		$\text{ccall } (\tau, ident, spine)$		C function call
		$\text{pcall } (name, spine)$		procedure call

$seq\_texpr$	$::=$ $  \quad tval$ $  \quad \mathbf{run} \, ident \, \overline{pval}_i^i$ $  \quad \mathbf{let} \, ident\_or\_pattern = pexpr \, \mathbf{in} \, texpr$ $  \quad \mathbf{let} \, ident\_or\_pattern:(y_1:\beta_1. term_1) = tpexpr \, \mathbf{in} \, texpr$  $  \quad \mathbf{let} \, \overline{ret\_pattern}_i^i = seq\_expr \, \mathbf{in} \, texpr$ $  \quad \mathbf{let} \, \overline{ret\_pattern}_i^i:ret = texpr_1 \, \mathbf{in} \, texpr_2$ $  \quad \mathbf{case} \, pval \, \mathbf{of} \,   \, texpr\_case\_branch_i^i \, \mathbf{end}$ $  \quad \mathbf{if} \, pval \, \mathbf{then} \, texpr_1 \, \mathbf{else} \, texpr_2$ $  \quad \mathbf{bound} \, [int](is\_texpr)$	   $\mathbf{bind} \, binders(ident\_or\_pattern) \, \mathbf{in} \, texpr$ $\mathbf{bind} \, binders(ident\_or\_pattern) \, \mathbf{in} \, texpr$ $\mathbf{bind} \, y_1 \, \mathbf{in} \, term_1$ $\mathbf{bind} \, binders(\overline{ret\_pattern}_i^i) \, \mathbf{in} \, texpr$ $\mathbf{bind} \, binders(\overline{ret\_pattern}_i^i) \, \mathbf{in} \, texpr_2$	sequential top-level (effectful) expressions (effectful) top-level values run from label pure let pure let  bind return patterns annotated bind return patterns pattern matching conditional limit scope of indet seq behaviour
$texpr\_case\_branch$	$::=$ $  \quad pattern \Rightarrow texpr$	$\mathbf{bind} \, binders(pattern) \, \mathbf{in} \, texpr$	top-level case expression branch top-level case expression branch
$is\_expr$	$::=$ $  \quad tval$ $  \quad \mathbf{memop} \, (mem\_op)$ $  \quad pol\_mem\_action$		indet seq (effectful) expressions (effectful) top-level values pointer op involving memory memory action
$is\_texpr$	$::=$ $  \quad \mathbf{let} \, \mathbf{weak} \, \overline{ret\_pattern}_i^i = is\_expr \, \mathbf{in} \, texpr$ $  \quad \mathbf{let} \, \mathbf{strong} \, \overline{ret\_pattern}_i^i = is\_expr \, \mathbf{in} \, texpr$	$\mathbf{bind} \, binders(\overline{ret\_pattern}_i^i) \, \mathbf{in} \, texpr$ $\mathbf{bind} \, binders(\overline{ret\_pattern}_i^i) \, \mathbf{in} \, texpr$	indet seq top-level (effectful) expressions weak sequencing strong sequencing
$texpr$	$::=$ $  \quad seq\_texpr$ $  \quad is\_texpr$ $  \quad \sigma(texpr)$	  M	top-level (effectful) expressions sequential (effectful) expressions indet seq (effectful) expressions simul-sub $\sigma$ in $texpr$
$arg$	$::=$		argument/function types

	$\Pi ident:\beta. arg$ $\forall ident:\beta. arg$ $res \multimap arg$ $term \supset arg$ $ret$ $\sigma(arg)$	M	simul-sub $\sigma$ in $arg$
$pure\_arg$	$::=$ $\Pi ident:\beta. pure\_arg$ $term \supset pure\_arg$ $pure\_ret$		pure argument/function types
$pure\_ret$	$::=$ $\Sigma ident:\beta. pure\_ret$ $term \wedge pure\_ret$ $\mathbf{I}$		pure return types
$\mathcal{C}$	$::=$ $\cdot$ $\mathcal{C}, ident:\beta$ $\overline{\mathcal{C}}_i^i$		computational var env
$\mathcal{L}$	$::=$ $\cdot$ $\overline{\mathcal{L}}_i^i$ $\mathcal{L}, ident:\beta$		logical var env
$\Phi$	$::=$ $\cdot$ $\Phi, term$		constraints env

		$\overline{\Phi}_i^i$	
$\mathcal{R}$	::=		resources env
		.	
		$\mathcal{R}, res$	
		$\mathcal{R}, ident:res$	
		$\overline{\mathcal{R}}_i^i$	
$\sigma, \psi$	::=		substitutions
		.	
		$spine\_elem/ident, \sigma$	
		$term/ident, \sigma$	
		$\overline{\sigma}_i^i$	
		$\sigma(\psi)$	M    apply $\sigma$ to all elements in $\psi$
$typing$	::=		
		<b>smt</b> ( $\Phi \Rightarrow term$ )	
		$ident:\beta \in \mathcal{C}$	
		$ident:\beta \in \mathcal{L}$	
		<b>struct</b> $tag \ \& \ \overline{member_i:\tau_i}^i \in \mathbf{Globals}$	
		$\overline{\mathcal{C}_i; \mathcal{L}_i; \Phi_i \vdash mem\_val_i \Rightarrow mem \beta_i}^i$	dependent on memory object model
$opsem$	::=		
		$\forall i < j. \mathbf{not} (pattern_i = pval \rightsquigarrow \sigma_i)$	
		<b>fresh</b> ( $mem\_ptr$ )	
		$term$	
		$pval:\beta$	
$formula$	::=		
		$judgement$	



	$ \begin{array}{ l} typing \\ opsem \\ term \equiv term' \\ name: pure\_arg \equiv \overline{x_i}^i \mapsto tpepr \in \mathbf{Globals} \\ name: arg \equiv \overline{x_i}^i \mapsto texpr \in \mathbf{Globals} \end{array} $	
$heap, h$	$ \begin{array}{ l} ::= \\ \cdot \\ h + \{points\_to\} \end{array} $	heaps
$object\_value\_jtype$	$ \begin{array}{ l} ::= \\ \mathcal{C}; \mathcal{L}; \Phi \vdash object\_value \Rightarrow \mathbf{obj} \beta \end{array} $	
$pval\_jtype$	$ \begin{array}{ l} ::= \\ \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \end{array} $	
$res\_jtype$	$ \begin{array}{ l} ::= \\ \Phi \vdash res \equiv res' \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash res\_term \Leftarrow res \end{array} $	
$spine\_jtype$	$ \begin{array}{ l} ::= \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret \end{array} $	
$pexpr\_jtype$	$ \begin{array}{ l} ::= \\ \mathcal{C}; \mathcal{L}; \Phi \vdash pexpr \Rightarrow ident:\beta. term \end{array} $	
$comp\_pattern\_jtype$	$ \begin{array}{ l} ::= \\ pattern:\beta \rightsquigarrow \mathcal{C} \mathbf{with} term \\ ident\_or\_pattern:\beta \rightsquigarrow \mathcal{C} \mathbf{with} term \end{array} $	

$res\_pattern\_jtype$	$::=$	$  \quad res\_pattern:res \rightsquigarrow \mathcal{L}; \Phi; \mathcal{R}$
$ret\_pattern\_jtype$	$::=$	$  \quad \overline{ret\_pattern_i}^i : ret \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}$
$tpval\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi \vdash tpval \Leftarrow ident:\beta. term$
$tpexpr\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi \vdash tpexpr \Leftarrow ident:\beta. term$
$action\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash mem\_action \Rightarrow ret$
$memop\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash mem\_op \Rightarrow ret$
$tval\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash tval \Leftarrow ret$
$seq\_expr\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash seq\_expr \Rightarrow ret$
$is\_expr\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_expr \Rightarrow ret$
$texpr\_jtype$	$::=$	$  \quad \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash seq\_texpr \Leftarrow ret$ $  \quad \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_texpr \Leftarrow ret$

		$\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{texpr} \Leftarrow \text{ret}$
$\text{subs\_jtype}$	$::=$	$\text{pattern} = \text{pval} \rightsquigarrow \sigma$   $\text{ident\_or\_pattern} = \text{pval} \rightsquigarrow \sigma$   $\text{res\_pattern} = \text{res\_term} \rightsquigarrow \sigma$   $\frac{\text{ret\_pattern}_i = \text{spine\_elem}_i}{\overline{x_i = \text{spine\_elem}_i}^i} \rightsquigarrow \sigma$   $\overline{x_i = \text{spine\_elem}_i}^i :: \text{arg} \gg \sigma; \text{ret}$
$\text{pure\_opsem\_jtype}$	$::=$	$\langle \text{pexpr} \rangle \longrightarrow \langle \text{pexpr}' \rangle$   $\langle \text{pexpr} \rangle \longrightarrow \langle \text{texpr}:(y:\beta. \text{term}) \rangle$   $\langle \text{texpr} \rangle \longrightarrow \langle \text{texpr}' \rangle$
$\text{opsem\_jtype}$	$::=$	$\langle h; \text{seq\_expr} \rangle \longrightarrow \langle h'; \text{texpr}:\text{ret} \rangle$   $\langle h; \text{seq\_texpr} \rangle \longrightarrow \langle h'; \text{texpr} \rangle$   $\langle h; \text{mem\_op} \rangle \longrightarrow \langle h'; \text{tval} \rangle$   $\langle h; \text{mem\_action} \rangle \longrightarrow \langle h'; \text{tval} \rangle$   $\langle h; \text{is\_expr} \rangle \longrightarrow \langle h'; \text{is\_expr}' \rangle$   $\langle h; \text{is\_texpr} \rangle \longrightarrow \langle h'; \text{texpr} \rangle$   $\langle h; \text{texpr} \rangle \longrightarrow \langle h'; \text{texpr}' \rangle$
$\text{lemma\_jtype}$	$::=$	$\overline{x_i}^i :: \text{arg} \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \mid \text{ret}$   $\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'$   $\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}')$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{object\_value} \Rightarrow \text{obj } \beta}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{mem\_int} \Rightarrow \text{obj integer}} \quad \text{Ty\_Pval\_Obj\_Int}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{mem\_ptr} \Rightarrow \text{obj loc}} \quad \text{TY\_PVAL\_OBJ\_PTR}$$

$$\frac{\overline{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{loaded\_value}_i \Rightarrow \beta^i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{array}(\text{loaded\_value}_i^i) \Rightarrow \text{obj array } \beta} \quad \text{TY\_PVAL\_OBJ\_ARR}$$

$$\frac{\frac{\text{struct tag} \ \& \ \overline{\text{member}_i: \tau_i^i} \in \text{Globals}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{mem\_val}_i \Rightarrow \text{mem } \beta_{\tau_i}^i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash (\text{struct tag})\{. \text{member}_i: \tau_i = \text{mem\_val}_i^i\} \Rightarrow \text{obj struct tag}} \quad \text{TY\_PVAL\_OBJ\_STRUCT}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow \beta}$$

$$\frac{x: \beta \in \mathcal{C}}{\mathcal{C}; \mathcal{L}; \Phi \vdash x \Rightarrow \beta} \quad \text{TY\_PVAL\_VAR\_COMP}$$

$$\frac{x: \beta \in \mathcal{L}}{\mathcal{C}; \mathcal{L}; \Phi \vdash x \Rightarrow \beta} \quad \text{TY\_PVAL\_VAR\_LOG}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{object\_value} \Rightarrow \text{obj } \beta}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{object\_value} \Rightarrow \beta} \quad \text{TY\_PVAL\_OBJ}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{object\_value} \Rightarrow \text{obj } \beta}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{specified object\_value} \Rightarrow \beta} \quad \text{TY\_PVAL\_LOADED}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{Unit} \Rightarrow \text{unit}} \quad \text{TY\_PVAL\_UNIT}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{True} \Rightarrow \text{bool}} \quad \text{TY\_PVAL\_TRUE}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{False} \Rightarrow \text{bool}} \quad \text{TY\_PVAL\_FALSE}$$

$$\frac{\overline{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{value}_i \Rightarrow \beta^i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \beta[\overline{\text{value}_i^i}] \Rightarrow \text{list } \beta} \quad \text{TY\_PVAL\_LIST}$$

$$\frac{\overline{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{value}_i \Rightarrow \beta_i^i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash (\overline{\text{value}_i^i}) \Rightarrow \overline{\beta_i^i}} \quad \text{TY\_PVAL\_TUPLE}$$

$$\frac{\text{smt}(\Phi \Rightarrow \text{false})}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{error}(\text{string}, \text{pval}) \Rightarrow \beta} \quad \text{TY\_PVAL\_ERROR}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{Nil } \beta() \Rightarrow \text{list } \beta} \quad \text{TY\_PVAL\_CTOR\_NIL}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_1 \Rightarrow \beta \\ \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_2 \Rightarrow \text{list } \beta \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{Cons}(\text{pval}_1, \text{pval}_2) \Rightarrow \text{list } \beta} \quad \text{TY\_PVAL\_CTOR\_CONS}$$

$$\frac{\overline{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_i \Rightarrow \beta_i^i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{Tuple}(\overline{\text{pval}_i^i}) \Rightarrow \overline{\beta_i^i}} \quad \text{TY\_PVAL\_CTOR\_TUPLE}$$

$$\frac{\overline{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_i \Rightarrow \beta^i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{Array}(\overline{\text{pval}_i^i}) \Rightarrow \text{array } \beta} \quad \text{TY\_PVAL\_CTOR\_ARRAY}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow \beta}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{Specified}(\text{pval}) \Rightarrow \beta} \quad \text{TY\_PVAL\_CTOR\_SPECIFIED}$$

$$\frac{\frac{\text{struct tag } \& \overline{\text{member}_i:\tau_i}^i \in \text{Globals}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_i \Rightarrow \beta_{\tau_i}^i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash (\text{struct tag})\{\text{.member}_i = \text{pval}_i\} \Rightarrow \text{struct tag}} \quad \text{TY\_PVAL\_STRUCT}$$

$$\boxed{\Phi \vdash \text{res} \equiv \text{res}'}$$

$$\frac{}{\Phi \vdash \text{emp} \equiv \text{emp}} \quad \text{TY\_RES\_EQ\_EMP}$$

$$\frac{\text{smt}(\Phi \Rightarrow (\text{term}_1 = \text{term}'_1) \wedge (\text{term}_2 = \text{term}'_2))}{\Phi \vdash \text{term}_1 \xrightarrow{\text{init}}_{\tau} \text{term}_2 \equiv \text{term}'_1 \xrightarrow{\text{init}}_{\tau} \text{term}'_2} \quad \text{TY\_RES\_EQ\_POINTSTO}$$

$$\frac{\begin{array}{l} \Phi \vdash \text{res}_1 \equiv \text{res}'_1 \\ \Phi \vdash \text{res}_2 \equiv \text{res}'_2 \end{array}}{\Phi \vdash \text{res}_1 * \text{res}_2 \equiv \text{res}'_1 * \text{res}'_2} \quad \text{TY\_RES\_EQ\_SEP\_CONJ}$$

$$\frac{\Phi \vdash \text{res} \equiv \text{res}'}{\Phi \vdash \exists \text{ident}:\beta. \text{res} \equiv \exists \text{ident}:\beta. \text{res}'} \quad \text{TY\_RES\_EQ\_EXISTS}$$

$$\frac{\begin{array}{l} \text{smt}(\Phi, \text{term} \Rightarrow \text{term}') \\ \text{smt}(\Phi, \text{term}' \Rightarrow \text{term}) \\ \Phi \vdash \text{res} \equiv \text{res}' \end{array}}{\Phi \vdash \text{term} \wedge \text{res} \equiv \text{term}' \wedge \text{res}'} \quad \text{TY\_RES\_EQ\_TERM}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{res\_term} \Leftarrow \text{res}}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \text{emp} \Leftarrow \text{emp}} \quad \text{TY\_RES\_EMP}$$

$$\frac{\Phi \vdash points\_to \equiv points\_to'}{\mathcal{C}; \mathcal{L}; \Phi; \cdot, points\_to \vdash points\_to' \Leftarrow points\_to'} \quad \text{TY\_RES\_POINTSTO}$$

$$\frac{\Phi \vdash res \equiv res'}{\mathcal{C}; \mathcal{L}; \Phi; \cdot, r:res \vdash r \Leftarrow res'} \quad \text{TY\_RES\_VAR}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_1 \vdash res\_term_1 \Leftarrow res_1 \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_2 \vdash res\_term_2 \Leftarrow res_2 \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_1, \mathcal{R}_2 \vdash \langle res\_term_1, res\_term_2 \rangle \Leftarrow res_1 * res_2} \quad \text{TY\_RES\_SEPCONJ}$$

$$\frac{\begin{array}{c} \text{smt}(\Phi \Rightarrow term) \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash res\_term \Leftarrow res \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash res\_term \Leftarrow term \wedge res} \quad \text{TY\_RES\_CONJ}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash res\_term_2 \Leftarrow pval/y, \cdot(res) \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{pack}(pval, res\_term_2) \Leftarrow \exists y:\beta. res} \quad \text{TY\_RES\_PACK}$$

$\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash ::ret \gg \cdot; ret} \quad \text{TY\_SPINE\_EMPTY}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash x = pval, \overline{x_i = spine\_elem_i}^i :: \Pi x:\beta. arg \gg pval/x, \sigma; ret} \quad \text{TY\_SPINE\_COMP}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash x = pval, \overline{x_i = spine\_elem_i}^i :: \forall x:\beta. arg \gg pval/x, \sigma; ret} \quad \text{TY\_SPINE\_LOG}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_1 \vdash \text{res\_term} \Leftarrow \text{res} \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_2 \vdash \overline{x_i = \text{spine\_elem}_i}^i :: \text{arg} \gg \sigma; \text{ret} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_1, \mathcal{R}_2 \vdash x = \text{res\_term}, \overline{x_i = \text{spine\_elem}_i}^i :: \text{res} \multimap \text{arg} \gg \text{res\_term}/x, \sigma; \text{ret}} \quad \text{TY\_SPINE\_RES}$$

$$\frac{\begin{array}{c} \text{smt}(\Phi \Rightarrow \text{term}) \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = \text{spine\_elem}_i}^i :: \text{arg} \gg \sigma; \text{ret} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = \text{spine\_elem}_i}^i :: \text{term} \supset \text{arg} \gg \sigma; \text{ret}} \quad \text{TY\_SPINE\_PHI}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pexpr} \Rightarrow \text{ident}:\beta. \text{term}}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow \beta}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow y:\beta. y = \text{pval}} \quad \text{TY\_PE\_VAL}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_1 \Rightarrow \text{loc} \\ \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_2 \Rightarrow \text{integer} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{array\_shift}(\text{pval}_1, \tau, \text{pval}_2) \Rightarrow y:\text{loc}. y = \text{pval}_1 +_{\text{ptr}} (\text{pval}_2 \times \text{size\_of}(\tau))} \quad \text{TY\_PE\_ARRAY\_SHIFT}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow \text{loc} \\ \text{struct } \text{tag} \ \& \ \overline{\text{member}_i:\tau_i}^i \in \text{Globals} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{member\_shift}(\text{pval}, \text{tag}, \text{member}_j) \Rightarrow y:\text{loc}. y = \text{pval} +_{\text{ptr}} \text{offset\_of}_{\text{tag}}(\text{member}_j)} \quad \text{TY\_PE\_MEMBER\_SHIFT}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow \text{bool}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{not}(\text{pval}) \Rightarrow y:\text{bool}. y = \neg \text{pval}} \quad \text{TY\_PE\_NOT}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_1 \Rightarrow \text{integer} \\ \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_2 \Rightarrow \text{integer} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval}_1 \ \text{binop}_{\text{arith}} \ \text{pval}_2 \Rightarrow y:\text{integer}. y = (\text{pval}_1 \ \text{binop}_{\text{arith}} \ \text{pval}_2)} \quad \text{TY\_PE\_ARITH\_BINOP}$$



$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \Rightarrow \text{integer} \\ \mathcal{C}; \mathcal{L}; \Phi \vdash pval_2 \Rightarrow \text{integer} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \text{ binop}_{rel} pval_2 \Rightarrow y:\text{bool}. y = (pval_1 \text{ binop}_{rel} pval_2)} \quad \text{TY\_PE\_REL\_BINOP}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \Rightarrow \text{bool} \\ \mathcal{C}; \mathcal{L}; \Phi \vdash pval_2 \Rightarrow \text{bool} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \text{ binop}_{bool} pval_2 \Rightarrow y:\text{bool}. y = (pval_1 \text{ binop}_{bool} pval_2)} \quad \text{TY\_PE\_BOOL\_BINOP}$$

$$\frac{\begin{array}{c} name: pure\_arg \equiv \overline{x_i}^i \mapsto tpe\_expr \in \text{Globals} \\ \mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash x_i = \overline{pval_i}^i :: pure\_arg \gg \sigma; \Sigma y:\beta. term \wedge \mathbf{I} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash name(\overline{pval_i}^i) \Rightarrow y:\beta. \sigma(term)} \quad \text{TY\_PE\_CALL}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \text{bool} \\ \text{smt}(\Phi \Rightarrow pval) \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{assert\_undef}(pval, UB\_name) \Rightarrow y:\text{unit}. y = \text{unit}} \quad \text{TY\_PE\_ASSERT\_UNDEF}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \text{bool}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{bool\_to\_integer}(pval) \Rightarrow y:\text{integer}. y = \text{if } pval \text{ then } 1 \text{ else } 0} \quad \text{TY\_PE\_BOOL\_TO\_INTEGER}$$

$$\frac{\begin{array}{c} \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \text{integer} \\ abbrev_1 \equiv \text{max\_int}_\tau - \text{min\_int}_\tau + 1 \\ abbrev_2 \equiv pval \text{ rem\_f } abbrev_1 \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{wrapI}(\tau, pval) \Rightarrow y:\beta. y = \text{if } abbrev_2 \leq \text{max\_int}_\tau \text{ then } abbrev_2 \text{ else } abbrev_2 - abbrev_1} \quad \text{TY\_PE\_WRAP I}$$

$pattern:\beta \rightsquigarrow \mathcal{C} \text{ with } term$

$$\frac{}{\cdot:\beta:\beta \rightsquigarrow \cdot \text{ with } \_} \quad \text{TY\_PAT\_COMP\_NO\_SYM\_ANNOT}$$

$$\frac{}{x:\beta:\beta \rightsquigarrow \cdot, x:\beta \textbf{with } x} \quad \text{TY\_PAT\_COMP\_SYM\_ANNOT}$$

$$\frac{}{\text{Nil } \beta():\text{list } \beta \rightsquigarrow \cdot \textbf{with nil}} \quad \text{TY\_PAT\_COMP\_NIL}$$

$$\frac{\begin{array}{l} \text{pattern}_1:\beta \rightsquigarrow \mathcal{C}_1 \textbf{with term}_1 \\ \text{pattern}_2:\text{list } \beta \rightsquigarrow \mathcal{C}_2 \textbf{with term}_2 \end{array}}{\text{Cons}(\text{pattern}_1, \text{pattern}_2):\text{list } \beta \rightsquigarrow \mathcal{C}_1, \mathcal{C}_2 \textbf{with term}_1 :: \text{term}_2} \quad \text{TY\_PAT\_COMP\_CONS}$$

$$\frac{\overline{\text{pattern}_i:\beta_i \rightsquigarrow \mathcal{C}_i \textbf{with term}_i^i}}{\text{Tuple}(\overline{\text{pattern}_i^i}):\overline{\beta_i^i} \rightsquigarrow \overline{\mathcal{C}_i^i} \textbf{with } (\overline{\text{term}_i^i})} \quad \text{TY\_PAT\_COMP\_TUPLE}$$

$$\frac{\overline{\text{pattern}_i:\beta \rightsquigarrow \mathcal{C}_i \textbf{with term}_i^i}}{\text{Array}(\overline{\text{pattern}_i^i}):\text{array } \beta \rightsquigarrow \overline{\mathcal{C}_i^i} \textbf{with } [\overline{\text{term}_i^i}] } \quad \text{TY\_PAT\_COMP\_ARRAY}$$

$$\frac{\text{pattern}:\beta \rightsquigarrow \mathcal{C} \textbf{with term}}{\text{Specified}(\text{pattern}):\beta \rightsquigarrow \mathcal{C} \textbf{with term}} \quad \text{TY\_PAT\_COMP\_SPECIFIED}$$

$$\boxed{\text{ident\_or\_pattern}:\beta \rightsquigarrow \mathcal{C} \textbf{with term}}$$

$$\frac{}{x:\beta \rightsquigarrow \cdot, x:\beta \textbf{with } x} \quad \text{TY\_PAT\_SYM\_OR\_PATTERN\_SYM}$$

$$\frac{\text{pattern}:\beta \rightsquigarrow \mathcal{C} \textbf{with term}}{\text{pattern}:\beta \rightsquigarrow \mathcal{C} \textbf{with term}} \quad \text{TY\_PAT\_SYM\_OR\_PATTERN\_PATTERN}$$

$$\boxed{\text{res\_pattern}:\text{res} \rightsquigarrow \mathcal{L}; \Phi; \mathcal{R}}$$

$$\frac{}{\text{emp}:\text{emp} \rightsquigarrow \cdot; \cdot; \cdot} \quad \text{TY\_PAT\_RES\_EMPTY}$$

$$\frac{}{points\_to:points\_to \rightsquigarrow \cdot; \cdot; \cdot, points\_to} \text{TY\_PAT\_RES\_POINTSTO}$$

$$\frac{}{r:res \rightsquigarrow \cdot; \cdot; \cdot, r:res} \text{TY\_PAT\_RES\_VAR}$$

$$\frac{\begin{array}{l} res\_pattern_1:res_1 \rightsquigarrow \mathcal{L}_1; \Phi_1; \mathcal{R}_1 \\ res\_pattern_2:res_2 \rightsquigarrow \mathcal{L}_2; \Phi_2; \mathcal{R}_2 \end{array}}{\langle res\_pattern_1, res\_pattern_2 \rangle:res_1 * res_2 \rightsquigarrow \mathcal{L}_1, \mathcal{L}_2; \Phi_1, \Phi_2; \mathcal{R}_1, \mathcal{R}_2} \text{TY\_PAT\_RES\_SEPCONJ}$$

$$\frac{res\_pattern:res \rightsquigarrow \mathcal{L}; \Phi; \mathcal{R}}{res\_pattern:term \wedge res \rightsquigarrow \mathcal{L}; \Phi, term; \mathcal{R}} \text{TY\_PAT\_RES\_CONJ}$$

$$\frac{res\_pattern:x/y, \cdot(res) \rightsquigarrow \mathcal{L}; \Phi; \mathcal{R}}{\text{pack}(x, res\_pattern):\exists y:\beta. res \rightsquigarrow \mathcal{L}, x:\beta; \Phi; \mathcal{R}} \text{TY\_PAT\_RES\_PACK}$$

$$\boxed{\overline{ret\_pattern_i}^i:ret \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}}$$

$$\frac{}{:I \rightsquigarrow \cdot; \cdot; \cdot; \cdot} \text{TY\_PAT\_RET\_EMPTY}$$

$$\frac{\begin{array}{l} ident\_or\_pattern:\beta \rightsquigarrow \mathcal{C}_1 \text{ with } term_1 \\ \overline{ret\_pattern_i}^i:term_1/y, \cdot(ret) \rightsquigarrow \mathcal{C}_2; \mathcal{L}_2; \Phi_2; \mathcal{R}_2 \end{array}}{\text{comp } ident\_or\_pattern, \overline{ret\_pattern_i}^i:\Sigma y:\beta. ret \rightsquigarrow \mathcal{C}_1, \mathcal{C}_2; \mathcal{L}_2; \Phi_2; \mathcal{R}_2} \text{TY\_PAT\_RET\_COMP}$$

$$\frac{\overline{ret\_pattern_i}^i:ret \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}}{\log y, \overline{ret\_pattern_i}^i:\exists y:\beta. ret \rightsquigarrow \mathcal{C}; \mathcal{L}, y:\beta; \Phi; \mathcal{R}} \text{TY\_PAT\_RET\_LOG}$$

$$\frac{\frac{\text{res\_pattern}:\text{res} \rightsquigarrow \mathcal{L}_1; \Phi_1; \mathcal{R}_1}{\overline{\text{ret\_pattern}_i}^i:\text{ret} \rightsquigarrow \mathcal{C}_2; \mathcal{L}_2; \Phi_2; \mathcal{R}_2}}{\text{res res\_pattern}, \overline{\text{ret\_pattern}_i}^i:\text{res} \otimes \text{ret} \rightsquigarrow \mathcal{C}_2; \mathcal{L}_1, \mathcal{L}_2; \Phi_1, \Phi_2; \mathcal{R}_1, \mathcal{R}_2} \quad \text{TY\_PAT\_RET\_RES}$$

$$\frac{\overline{\text{ret\_pattern}_i}^i:\text{ret} \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}}{\overline{\text{ret\_pattern}_i}^i:\text{term} \wedge \text{ret} \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi, \text{term}; \mathcal{R}} \quad \text{TY\_PAT\_RET\_PHI}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{tpval} \Leftarrow \text{ident}:\beta. \text{term}}$$

$$\frac{\text{smt}(\Phi \Rightarrow \text{false})}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{undef } UB\_name \Leftarrow y:\beta. \text{term}} \quad \text{TY\_TPVAL\_UNDEF}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow \beta \\ \text{smt}(\Phi \Rightarrow \text{pval}/y, \cdot(\text{term})) \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{done pval} \Leftarrow y:\beta. \text{term}} \quad \text{TY\_TPVAL\_DONE}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{tpexpr} \Leftarrow \text{ident}:\beta. \text{term}}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pval} \Rightarrow \text{bool} \\ \mathcal{C}; \mathcal{L}; \Phi, \text{pval} = \text{true} \vdash \text{tpexpr}_1 \Leftarrow y:\beta. \text{term} \\ \mathcal{C}; \mathcal{L}; \Phi, \text{pval} = \text{false} \vdash \text{tpexpr}_2 \Leftarrow y:\beta. \text{term} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{if pval then tpexpr}_1 \text{ else tpexpr}_2 \Leftarrow y:\beta. \text{term}} \quad \text{TY\_TPE\_IF}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash \text{pexpr} \Rightarrow y_1:\beta_1. \text{term}_1 \\ \text{ident\_or\_pattern}:\beta_1 \rightsquigarrow \mathcal{C}_1 \text{ with term} \\ \mathcal{C}, \mathcal{C}_1; \mathcal{L}; \Phi, \text{term}/y_1, \cdot(\text{term}_1) \vdash \text{tpexpr} \Leftarrow y_2:\beta_2. \text{term}_2 \end{array}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \text{let ident\_or\_pattern} = \text{pexpr in tpexpr} \Leftarrow y_2:\beta_2. \text{term}_2} \quad \text{TY\_TPE\_LET}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash tpepr_1 \Leftarrow y_1:\beta_1. term_1 \\
ident\_or\_pattern:\beta_1 \rightsquigarrow \mathcal{C}_1 \text{ with } term \\
\mathcal{C}, \mathcal{C}_1; \mathcal{L}; \Phi, term/y_1, \cdot (term_1) \vdash tpepr \Leftarrow y_2:\beta_2. term_2 \\
\hline
\mathcal{C}; \mathcal{L}; \Phi \vdash \text{let } ident\_or\_pattern:(y_1:\beta_1. term_1) = tpepr_1 \text{ in } tpepr_2 \Leftarrow y_2:\beta_2. term_2
\end{array}
\quad \text{TY\_TPE\_LET}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta_1 \\
\overline{pattern_i:\beta_1 \rightsquigarrow \mathcal{C}_i \text{ with } term_i}^i \\
\mathcal{C}, \mathcal{C}_i; \mathcal{L}; \Phi, term_i = pval \vdash tpepr_i \Leftarrow y_2:\beta_2. term_2^i \\
\hline
\mathcal{C}; \mathcal{L}; \Phi \vdash \text{case } pval \text{ of } \overline{pattern_i \Rightarrow tpepr_i}^i \text{ end } \Leftarrow y_2:\beta_2. term_2
\end{array}
\quad \text{TY\_TPE\_CASE}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash mem\_action \Rightarrow ret}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \text{integer} \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \text{create } (pval, \tau) \Rightarrow \Sigma y_p:\text{loc. representable } (\tau*, y_p) \wedge \text{alignedI } (pval, y_p) \wedge \exists y:\beta_\tau. y_p \overset{\times}{\mapsto}_\tau y \otimes \mathbf{I}
\end{array}
\quad \text{TY\_ACTION\_CREATE}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_0 \Rightarrow \text{loc} \\
\text{smt } (\Phi \Rightarrow pval_0 = pval_1) \\
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash pval_1 \overset{\checkmark}{\mapsto}_\tau pval_2 \Leftarrow pval_1 \overset{\checkmark}{\mapsto}_\tau pval_2 \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{load } (\tau, pval_0, -, pval_1 \overset{\checkmark}{\mapsto}_\tau pval_2) \Rightarrow \Sigma y:\beta_\tau. y = pval_2 \wedge pval_1 \overset{\checkmark}{\mapsto}_\tau pval_2 \otimes \mathbf{I}
\end{array}
\quad \text{TY\_ACTION\_LOAD}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_0 \Rightarrow \text{loc} \\
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \Rightarrow \beta_\tau \\
\text{smt } (\Phi \Rightarrow \text{representable } (\tau, pval_1)) \\
\text{smt } (\Phi \Rightarrow pval_2 = pval_0) \\
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash pval_2 \mapsto_\tau - \Leftarrow pval_2 \mapsto_\tau - \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{store } (-, \tau, pval_0, pval_1, -, pval_2 \mapsto_\tau -) \Rightarrow \Sigma -: \text{unit. } pval_2 \overset{\checkmark}{\mapsto}_\tau pval_1 \otimes \mathbf{I}
\end{array}
\quad \text{TY\_ACTION\_STORE}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_0 \Rightarrow \text{loc} \\
\text{smt}(\Phi \Rightarrow pval_0 = pval_1) \\
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash pval_1 \mapsto_{\tau} - \Leftarrow pval_1 \mapsto_{\tau} - \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{kill}(\text{static } \tau, pval_0, pval_1 \mapsto_{\tau} -) \Rightarrow \Sigma \_:\text{unit}. \mathbf{I}
\end{array}
\quad \text{TY\_ACTION\_KILL\_STATIC}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{mem\_op} \Rightarrow \text{ret}}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \Rightarrow \text{loc} \\
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_2 \Rightarrow \text{loc} \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash pval_1 \text{ binop}_{rel} pval_2 \Rightarrow \Sigma y:\text{bool}. y = (pval_1 \text{ binop}_{rel} pval_2) \wedge \mathbf{I}
\end{array}
\quad \text{TY\_MEMOP\_REL\_BINOP}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \text{loc} \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \text{intFromPtr}(\tau_1, \tau_2, pval) \Rightarrow \Sigma y:\text{integer}. y = \text{cast\_ptr\_to\_int } pval \wedge \mathbf{I}
\end{array}
\quad \text{TY\_MEMOP\_INTFROMPTR}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \text{integer} \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \text{ptrFromInt}(\tau_1, \tau_2, pval) \Rightarrow \Sigma y:\text{loc}. y = \text{cast\_int\_to\_ptr } pval \wedge \mathbf{I}
\end{array}
\quad \text{TY\_MEMOP\_PTRFROMINT}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_0 \Rightarrow \text{loc} \\
\text{smt}(\Phi \Rightarrow pval_1 = pval_0) \\
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash pval_1 \check{\mapsto}_{\tau} - \Leftarrow pval_1 \check{\mapsto}_{\tau} - \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{ptrValidForDeref}(\tau, pval_0, pval_1 \check{\mapsto}_{\tau} -) \Rightarrow \Sigma y:\text{bool}. y = \text{aligned}(\tau, pval_1) \wedge pval_1 \check{\mapsto}_{\tau} - \otimes \mathbf{I}
\end{array}
\quad \text{TY\_MEMOP\_PTRVALIDFORDEREF}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \Rightarrow \text{loc} \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \text{ptrWellAligned}(\tau, pval) \Rightarrow \Sigma y:\text{bool}. y = \text{aligned}(\tau, pval) \wedge \mathbf{I}
\end{array}
\quad \text{TY\_MEMOP\_PTRWELLALIGNED}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_1 \Rightarrow \text{loc} \\
\mathcal{C}; \mathcal{L}; \Phi \vdash pval_2 \Rightarrow \text{integer} \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \text{ptrArrayShift}(pval_1, \tau, pval_2) \Rightarrow \Sigma y:\text{loc}. y = pval_1 +_{\text{ptr}} (pval_2 \times \text{size\_of}(\tau)) \wedge \mathbf{I}
\end{array}
\quad \text{TY\_MEMOP\_PTRARRAYSHIFT}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash tval \Leftarrow ret}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \mathbf{done} \Leftarrow \mathbf{I}} \text{TY\_TVAL\_I}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{done} \overline{spine\_elem_i}^i \Leftarrow pval/y, \cdot (ret) \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{done} pval, \overline{spine\_elem_i}^i \Leftarrow \Sigma y:\beta. ret} \text{TY\_TVAL\_COMP}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{done} \overline{spine\_elem_i}^i \Leftarrow pval/y, \cdot (ret) \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{done} pval, \overline{spine\_elem_i}^i \Leftarrow \exists y:\beta. ret} \text{TY\_TVAL\_LOG}$$

$$\frac{\begin{array}{l} \mathbf{smt}(\Phi \Rightarrow term) \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{done} spine \Leftarrow ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{done} spine \Leftarrow term \wedge ret} \text{TY\_TVAL\_PHI}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_1 \vdash res\_term \Leftarrow res \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_2 \vdash \mathbf{done} \overline{spine\_elem_i}^i \Leftarrow ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_1, \mathcal{R}_2 \vdash \mathbf{done} res\_term, \overline{spine\_elem_i}^i \Leftarrow res \otimes ret} \text{TY\_TVAL\_RES}$$

$$\frac{\mathbf{smt}(\Phi \Rightarrow \mathbf{false})}{\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \mathbf{undef} UB\_name \Leftarrow ret} \text{TY\_TVAL\_UNDEF}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash seq\_expr \Rightarrow ret}$$

$$\frac{\begin{array}{l} ident:arg \equiv \overline{x_i}^i \mapsto texpr \in \mathbf{Globals} \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{ccall}(\tau, ident, \overline{spine\_elem_i}^i) \Rightarrow \sigma(ret)} \text{TY\_SEQ\_E\_CCALL}$$

$$\frac{\begin{array}{l} name:arg \equiv \overline{x_i}^i \mapsto texpr \in \mathbf{Globals} \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{pcall}(name, \overline{spine\_elem_i}^i) \Rightarrow \sigma(ret)} \quad \mathbf{TY\_SEQ\_E\_PROC}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_expr \Rightarrow ret}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash mem\_op \Rightarrow ret}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{memop}(mem\_op) \Rightarrow ret} \quad \mathbf{TY\_IS\_E\_MEMOP}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash mem\_action \Rightarrow ret}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash mem\_action \Rightarrow ret} \quad \mathbf{TY\_IS\_E\_ACTION}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash mem\_action \Rightarrow ret}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{neg} mem\_action \Rightarrow ret} \quad \mathbf{TY\_IS\_E\_NEG\_ACTION}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash seq\_texpr \Leftarrow ret}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash tval \Leftarrow ret}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash tval \Leftarrow ret} \quad \mathbf{TY\_SEQ\_TE\_TVAL}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash pexpr \Rightarrow y:\beta. term \\ ident\_or\_pattern:\beta \rightsquigarrow \mathcal{C}_1 \mathbf{with} term_1 \\ \mathcal{C}, \mathcal{C}_1; \mathcal{L}; \Phi, term_1/y, \cdot(term); \mathcal{R} \vdash texpr \Leftarrow ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{let} ident\_or\_pattern = pexpr \mathbf{in} texpr \Leftarrow ret} \quad \mathbf{TY\_SEQ\_TE\_LETP}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi \vdash tpexpr \Leftarrow y:\beta. term \\ ident\_or\_pattern:\beta \rightsquigarrow \mathcal{C}_1 \mathbf{with} term_1 \\ \mathcal{C}, \mathcal{C}_1; \mathcal{L}; \Phi, term_1/y, \cdot(term); \mathcal{R} \vdash texpr \Leftarrow ret \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{let} ident\_or\_pattern:(y:\beta. term) = tpexpr \mathbf{in} texpr \Leftarrow ret} \quad \mathbf{TY\_SEQ\_TE\_LETPT}$$



$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}' \vdash seq\_expr \Rightarrow ret_1 \\
\overline{ret\_pattern_i}^i : ret_1 \rightsquigarrow \mathcal{C}_1; \mathcal{L}_1; \Phi_1; \mathcal{R}_1 \\
\mathcal{C}, \mathcal{C}_1; \mathcal{L}, \mathcal{L}_1; \Phi, \Phi_1; \mathcal{R}, \mathcal{R}_1 \vdash texpr \Leftarrow ret_2 \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}', \mathcal{R} \vdash \mathbf{let} \overline{ret\_pattern_i}^i = seq\_expr \mathbf{in} texpr \Leftarrow ret_2
\end{array}
\quad \text{TY\_SEQ\_TE\_LET}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}' \vdash texpr_1 \Leftarrow ret_1 \\
\overline{ret\_pattern_i}^i : ret_1 \rightsquigarrow \mathcal{C}_1; \mathcal{L}_1; \Phi_1; \mathcal{R}_1 \\
\mathcal{C}, \mathcal{C}_1; \mathcal{L}, \mathcal{L}_1; \Phi, \Phi_1; \mathcal{R}, \mathcal{R}_1 \vdash texpr_2 \Leftarrow ret_2 \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}', \mathcal{R} \vdash \mathbf{let} \overline{ret\_pattern_i}^i : ret_1 = texpr_1 \mathbf{in} texpr_2 \Leftarrow ret_2
\end{array}
\quad \text{TY\_SEQ\_TE\_LET}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta_1 \\
\overline{pattern_i}^i : \beta_1 \rightsquigarrow \mathcal{C}_i \mathbf{with} term_i \\
\mathcal{C}, \mathcal{C}_i; \mathcal{L}; \Phi, term_i = pval; \mathcal{R} \vdash texpr_i \Leftarrow ret^i \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{case} pval \mathbf{of} \mid \overline{pattern_i}^i \Rightarrow texpr_i \mathbf{end} \Leftarrow ret
\end{array}
\quad \text{TY\_SEQ\_TE\_CASE}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \mathbf{bool} \\
\mathcal{C}; \mathcal{L}; \Phi, pval = \mathbf{true}; \mathcal{R} \vdash texpr_1 \Leftarrow ret \\
\mathcal{C}; \mathcal{L}; \Phi, pval = \mathbf{false}; \mathcal{R} \vdash texpr_2 \Leftarrow ret \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{if} pval \mathbf{then} texpr_1 \mathbf{else} texpr_2 \Leftarrow ret
\end{array}
\quad \text{TY\_SEQ\_TE\_IF}$$

$$\begin{array}{c}
ident:arg \equiv \overline{x_i}^i \mapsto texpr \in \mathbf{Globals} \\
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \overline{x_i}^i = pval_i^i :: arg \gg \sigma; \mathbf{false} \wedge \mathbf{I} \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash \mathbf{run} ident \overline{pval_i}^i \Leftarrow \mathbf{false} \wedge \mathbf{I}
\end{array}
\quad \text{TY\_SEQ\_TE\_RUN}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_texpr \Leftarrow ret \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathbf{bound} [int](is\_texpr) \Leftarrow ret
\end{array}
\quad \text{TY\_SEQ\_TE\_BOUND}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_texpr \Leftarrow ret}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_expr \Rightarrow ret_1 \\
\overline{ret\_pattern_i}^i : ret_1 \rightsquigarrow \mathcal{C}_1; \mathcal{L}_1; \Phi_1; \mathcal{R}_1 \\
\mathcal{C}, \mathcal{C}_1; \mathcal{L}, \mathcal{L}_1; \Phi, \Phi_1; \mathcal{R}, \mathcal{R}_1 \vdash texpr \Leftarrow ret_2 \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \text{let strong } \overline{ret\_pattern_i}^i = is\_expr \text{ in } texpr \Leftarrow ret_2
\end{array}
\quad \text{TY\_IS\_TE\_LETS}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash texpr \Leftarrow ret}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_texpr \Leftarrow ret \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash is\_texpr \Leftarrow ret
\end{array}
\quad \text{TY\_TE\_IS}$$

$$\begin{array}{c}
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash seq\_texpr \Leftarrow ret \\
\hline
\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash seq\_texpr \Leftarrow ret
\end{array}
\quad \text{TY\_TE\_SEQ}$$

$$\boxed{pattern = pval \rightsquigarrow \sigma}$$

$$\frac{}{\therefore \_ = pval \rightsquigarrow \cdot} \quad \text{SUBS\_DECONS\_VALUE\_NO\_SYM\_ANNOT}$$

$$\frac{}{x: \_ = pval \rightsquigarrow pval/x, \cdot} \quad \text{SUBS\_DECONS\_VALUE\_SYM\_ANNOT}$$

$$\begin{array}{c}
pattern_1 = pval_1 \rightsquigarrow \sigma_1 \\
pattern_2 = pval_2 \rightsquigarrow \sigma_2 \\
\hline
\text{Cons}(pattern_1, pattern_2) = \text{Cons}(pval_1, pval_2) \rightsquigarrow \sigma_1, \sigma_2
\end{array}
\quad \text{SUBS\_DECONS\_VALUE\_CONS}$$

$$\begin{array}{c}
\overline{pattern_i}^i = \overline{pval_i}^i \rightsquigarrow \sigma_i \\
\hline
\text{Tuple}(\overline{pattern_i}^i) = \text{Tuple}(\overline{pval_i}^i) \rightsquigarrow \overline{\sigma_i}^i
\end{array}
\quad \text{SUBS\_DECONS\_VALUE\_TUPLE}$$

$$\frac{\overline{pattern_i = pval_i \rightsquigarrow \sigma_i^i}}{\text{Array}(\overline{pattern_i^i}) = \text{Array}(\overline{pval_i^i}) \rightsquigarrow \overline{\sigma_i^i}} \quad \text{SUBS\_DECONS\_VALUE\_ARRAY}$$

$$\frac{pattern = pval \rightsquigarrow \sigma}{\text{Specified}(pattern) = pval \rightsquigarrow \sigma} \quad \text{SUBS\_DECONS\_VALUE\_SPECIFIED}$$

$$\boxed{ident\_or\_pattern = pval \rightsquigarrow \sigma}$$

$$\frac{}{x = pval \rightsquigarrow pval/x, \cdot} \quad \text{SUBS\_DECONS\_VALUE\_SYM}$$

$$\frac{pattern = pval \rightsquigarrow \sigma}{pattern = pval \rightsquigarrow \sigma} \quad \text{SUBS\_DECONS\_VALUE\_PATTERN}$$

$$\boxed{res\_pattern = res\_term \rightsquigarrow \sigma}$$

$$\frac{}{\mathbf{emp} = \mathbf{emp} \rightsquigarrow \cdot} \quad \text{SUBS\_DECONS\_RES\_EMP}$$

$$\frac{}{pt = pt \rightsquigarrow \cdot} \quad \text{SUBS\_DECONS\_RES\_POINTS\_TO}$$

$$\frac{}{ident = res\_term \rightsquigarrow res\_term/ident, \cdot} \quad \text{SUBS\_DECONS\_RES\_VAR}$$

$$\frac{\begin{array}{l} res\_pattern_1 = res\_term_1 \rightsquigarrow \sigma_1 \\ res\_pattern_2 = res\_term_2 \rightsquigarrow \sigma_2 \end{array}}{\langle res\_pattern_1, res\_pattern_2 \rangle = \langle res\_term_1, res\_term_2 \rangle \rightsquigarrow \sigma_1, \sigma_2} \quad \text{SUBS\_DECONS\_RES\_PAIR}$$

$$\frac{res\_pattern = res\_term \rightsquigarrow \sigma}{\mathbf{pack}(ident, res\_pattern) = \mathbf{pack}(pval, res\_term) \rightsquigarrow pval/ident, \sigma} \quad \text{SUBS\_DECONS\_RES\_PACK}$$

$$\boxed{\overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow \sigma}$$

$$\frac{}{\rightsquigarrow \cdot} \quad \text{SUBS\_DECONS\_RET\_EMPTY}$$

$$\frac{\frac{ident\_or\_pattern = pval \rightsquigarrow \sigma}{\overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow \psi}}{\text{comp } ident\_or\_pattern = pval, \overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow \sigma, \psi} \quad \text{SUBS\_DECONS\_RET\_COMP}$$

$$\frac{\overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow \psi}{\text{log } ident = pval, \overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow pval/ident, \psi} \quad \text{SUBS\_DECONS\_RET\_LOG}$$

$$\frac{\frac{res\_pattern = res\_term \rightsquigarrow \sigma}{\overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow \psi}}{\text{res } res\_pattern = res\_term, \overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow \sigma, \psi} \quad \text{SUBS\_DECONS\_RET\_RES}$$

$$\boxed{x_i = spine\_elem_i^i :: arg \gg \sigma; ret}$$

$$\frac{}{::ret \gg \cdot; ret} \quad \text{SUBS\_DECONS\_ARG\_EMPTY}$$

$$\frac{\overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret}{x = pval, \overline{x_i = spine\_elem_i}^i :: \Pi x:\beta. arg \gg pval/x, \sigma; ret} \quad \text{SUBS\_DECONS\_ARG\_COMP}$$

$$\frac{\overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret}{x = pval, \overline{x_i = spine\_elem_i}^i :: \forall x:\beta. arg \gg pval/x, \sigma; ret} \quad \text{SUBS\_DECONS\_ARG\_LOG}$$

$$\frac{\overline{x_i = spine\_elem_i^i :: arg \gg \sigma; ret}}{x = res\_term, \overline{x_i = spine\_elem_i^i :: res \multimap arg \gg res\_term/x, \sigma; ret}} \quad \text{SUBS\_DECONS\_ARG\_RES}$$

$$\frac{\overline{x_i = spine\_elem_i^i :: arg \gg \sigma; ret}}{\overline{x_i = spine\_elem_i^i :: term \supset arg \gg \sigma; ret}} \quad \text{SUBS\_DECONS\_ARG\_PHI}$$

$$\boxed{\langle pexpr \rangle \longrightarrow \langle pexpr' \rangle}$$

$$\frac{mem\_ptr' \equiv mem\_ptr +_{\text{ptr}} mem\_int \times \text{size\_of}(\tau)}{\langle \text{array\_shift}(mem\_ptr, \tau, mem\_int) \rangle \longrightarrow \langle mem\_ptr' \rangle} \quad \text{OP\_PE\_PE\_ARRAYSHIFT}$$

$$\frac{mem\_ptr' \equiv mem\_ptr +_{\text{ptr}} \text{offset\_of}_{tag}(member)}{\langle \text{member\_shift}(mem\_ptr, tag, member) \rangle \longrightarrow \langle mem\_ptr' \rangle} \quad \text{OP\_PE\_PE\_MEMBERSHIFT}$$

$$\frac{}{\langle \text{not}(\text{True}) \rangle \longrightarrow \langle \text{False} \rangle} \quad \text{OP\_PE\_PE\_NOT\_TRUE}$$

$$\frac{}{\langle \text{not}(\text{False}) \rangle \longrightarrow \langle \text{True} \rangle} \quad \text{OP\_PE\_PE\_NOT\_FALSE}$$

$$\frac{mem\_int \equiv mem\_int_1 \text{ binop}_{arith} mem\_int_2}{\langle mem\_int_1 \text{ binop}_{arith} mem\_int_2 \rangle \longrightarrow \langle mem\_int \rangle} \quad \text{OP\_PE\_PE\_ARITH\_BINOP}$$

$$\frac{bool\_value \equiv mem\_int_1 \text{ binop}_{rel} mem\_int_2}{\langle mem\_int_1 \text{ binop}_{rel} mem\_int_2 \rangle \longrightarrow \langle bool\_value \rangle} \quad \text{OP\_PE\_PE\_REL\_BINOP}$$

$$\frac{bool\_value \equiv bool\_value_1 \text{ binop}_{bool} bool\_value_2}{\langle bool\_value_1 \text{ binop}_{bool} bool\_value_2 \rangle \longrightarrow \langle bool\_value \rangle} \quad \text{OP\_PE\_PE\_BOOL\_BINOP}$$

$$\frac{}{\langle \text{assert\_undef}(\text{True}, UB\_name) \rangle \longrightarrow \langle \text{Unit} \rangle} \quad \text{OP\_PE\_PE\_ASSERT\_UNDEF}$$

$$\frac{}{\langle \text{bool\_to\_integer}(\text{True}) \rangle \longrightarrow \langle 1 \rangle} \quad \text{OP\_PE\_PE\_BOOL\_TO\_INTEGER\_TRUE}$$

$$\frac{}{\langle \text{bool\_to\_integer}(\text{False}) \rangle \longrightarrow \langle 0 \rangle} \quad \text{OP\_PE\_PE\_BOOL\_TO\_INTEGER\_FALSE}$$

$$\frac{\begin{array}{l} abbrev_1 \equiv \text{max\_int}_\tau - \text{min\_int}_\tau + 1 \\ abbrev_2 \equiv pval \text{ rem\_f } abbrev_1 \\ mem\_int' \equiv \text{if } abbrev_2 \leq \text{max\_int}_\tau \text{ then } abbrev_2 \text{ else } abbrev_2 - abbrev_1 \end{array}}{\langle \text{wrapI}(\tau, mem\_int) \rangle \longrightarrow \langle mem\_int' \rangle} \quad \text{OP\_PE\_PE\_WRAP\_I}$$

$$\boxed{\langle pexpr \rangle \longrightarrow \langle tpepr:(y:\beta. term) \rangle}$$

$$\frac{\begin{array}{l} name: pure\_arg \equiv \overline{x_i}^i \mapsto tpepr \in \text{Globals} \\ \overline{x_i = pval_i}^i :: pure\_arg \gg \sigma; \Sigma y:\beta. term \wedge I \end{array}}{\langle name(\overline{pval_i}^i) \rangle \longrightarrow \langle \sigma(tpepr):(y:\beta. \sigma(term)) \rangle} \quad \text{OP\_PE\_TPE\_CALL}$$

$$\boxed{\langle tpepr \rangle \longrightarrow \langle tpepr' \rangle}$$

$$\frac{\begin{array}{l} pattern_j = pval \rightsquigarrow \sigma_j \\ \forall i < j. \text{not } (pattern_i = pval \rightsquigarrow \sigma_i) \end{array}}{\langle \text{case } pval \text{ of } \overline{pattern_i \Rightarrow tpepr_i}^i \text{ end} \rangle \longrightarrow \langle \sigma_j(tpepr_j) \rangle} \quad \text{OP\_TPE\_TPE\_CASE}$$

$$\frac{ident\_or\_pattern = pval \rightsquigarrow \sigma}{\langle \text{let } ident\_or\_pattern = pval \text{ in } tpepr \rangle \longrightarrow \langle \sigma(tpepr) \rangle} \quad \text{OP\_TPE\_TPE\_LET\_SUB}$$

$$\frac{\langle pexpr \rangle \longrightarrow \langle pexpr' \rangle}{\langle \text{let } ident\_or\_pattern = pexpr \text{ in } tpepr \rangle \longrightarrow \langle \text{let } ident\_or\_pattern = pexpr' \text{ in } tpepr \rangle} \quad \text{OP\_TPE\_TPE\_LET\_LET}$$

$$\frac{\langle pexpr \rangle \longrightarrow \langle texpr_1:(y:\beta. term) \rangle}{\langle \text{let } ident\_or\_pattern = pexpr \text{ in } texpr_2 \rangle \longrightarrow \langle \text{let } ident\_or\_pattern:(y:\beta. term) = texpr_1 \text{ in } texpr_2 \rangle} \quad \text{OP\_TPE\_TPE\_LET\_LET}$$

$$\frac{ident\_or\_pattern = pval \rightsquigarrow \sigma}{\langle \text{let } ident\_or\_pattern:(y:\beta. term) = \text{done } pval \text{ in } texpr \rangle \longrightarrow \langle \sigma(texpr) \rangle} \quad \text{OP\_TPE\_TPE\_LET\_SUB}$$

$$\frac{\langle texpr_1 \rangle \longrightarrow \langle texpr'_1 \rangle}{\langle \text{let } ident\_or\_pattern:(y:\beta. term) = texpr_1 \text{ in } texpr_2 \rangle \longrightarrow \langle \text{let } ident\_or\_pattern:(y:\beta. term) = texpr'_1 \text{ in } texpr_2 \rangle} \quad \text{OP\_TPE\_TPE\_LET\_LET}$$

$$\frac{}{\langle \text{if True then } texpr_1 \text{ else } texpr_2 \rangle \longrightarrow \langle texpr_1 \rangle} \quad \text{OP\_TPE\_TPE\_IF\_TRUE}$$

$$\frac{}{\langle \text{if False then } texpr_1 \text{ else } texpr_2 \rangle \longrightarrow \langle texpr_2 \rangle} \quad \text{OP\_TPE\_TPE\_IF\_FALSE}$$

$$\boxed{\langle h; seq\_expr \rangle \longrightarrow \langle h'; texpr:ret \rangle}$$

$$\frac{\begin{array}{l} ident:arg \equiv \overline{x_i}^i \mapsto texpr \in \text{Globals} \\ \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret \end{array}}{\langle h; ccall(\tau, ident, \overline{spine\_elem_i}^i) \rangle \longrightarrow \langle h; \sigma(texpr):\sigma(ret) \rangle} \quad \text{OP\_SE\_TE\_CCALL}$$

$$\frac{\begin{array}{l} name:arg \equiv \overline{x_i}^i \mapsto texpr \in \text{Globals} \\ \overline{x_i = spine\_elem_i}^i :: arg \gg \sigma; ret \end{array}}{\langle h; pcall(name, \overline{spine\_elem_i}^i) \rangle \longrightarrow \langle h; \sigma(texpr):\sigma(ret) \rangle} \quad \text{OP\_SE\_TE\_PCALL}$$

$$\boxed{\langle h; seq\_texpr \rangle \longrightarrow \langle h'; texpr \rangle}$$

$$\frac{\begin{array}{l} ident:arg \equiv \overline{x_i}^i \mapsto texpr \in \text{Globals} \\ \overline{x_i = pval_i}^i :: arg \gg \sigma; \text{false} \wedge \text{I} \end{array}}{\langle h; \text{run } ident \overline{pval_i}^i \rangle \longrightarrow \langle h; \sigma(texpr) \rangle} \quad \text{OP\_STE\_TE\_RUN}$$

$$\frac{\begin{array}{c} pattern_j = pval \rightsquigarrow \sigma_j \\ \forall i < j. \text{not } (pattern_i = pval \rightsquigarrow \sigma_i) \end{array}}{\langle h; \text{case } pval \text{ of } \mid pattern_i \Rightarrow texpr_i^i \text{ end} \rangle \longrightarrow \langle h; \sigma_j(texpr_j) \rangle} \quad \text{OP\_STE\_TE\_CASE}$$

$$\frac{ident\_or\_pattern = pval \rightsquigarrow \sigma}{\langle h; \text{let } ident\_or\_pattern = pval \text{ in } texpr \rangle \longrightarrow \langle h; \sigma(texpr) \rangle} \quad \text{OP\_STE\_TE\_LETP\_SUB}$$

$$\frac{\langle pexpr \rangle \longrightarrow \langle pexpr' \rangle}{\langle h; \text{let } ident\_or\_pattern = pexpr \text{ in } texpr \rangle \longrightarrow \langle h; \text{let } ident\_or\_pattern = pexpr' \text{ in } texpr \rangle} \quad \text{OP\_STE\_TE\_LETP\_LETP}$$

$$\frac{\langle pexpr \rangle \longrightarrow \langle texpr:(y:\beta. term) \rangle}{\langle h; \text{let } ident\_or\_pattern = pexpr \text{ in } texpr \rangle \longrightarrow \langle h; \text{let } ident\_or\_pattern:(y:\beta. term) = texpr \text{ in } texpr \rangle} \quad \text{OP\_STE\_TE\_LETP\_LETP}$$

$$\frac{ident\_or\_pattern = pval \rightsquigarrow \sigma}{\langle h; \text{let } ident\_or\_pattern:(y:\beta. term) = \text{done } pval \text{ in } texpr \rangle \longrightarrow \langle h; \sigma(texpr) \rangle} \quad \text{OP\_STE\_TE\_LETP\_SUB}$$

$$\frac{\langle texpr \rangle \longrightarrow \langle texpr' \rangle}{\langle h; \text{let } ident\_or\_pattern:(y:\beta. term) = texpr \text{ in } texpr \rangle \longrightarrow \langle h; \text{let } ident\_or\_pattern:(y:\beta. term) = texpr' \text{ in } texpr \rangle} \quad \text{OP\_STE\_TE\_LETP\_LETP}$$

$$\frac{\overline{ret\_pattern_i = spine\_elem_i^i} \rightsquigarrow \sigma}{\langle h; \text{let } \overline{ret\_pattern_i^i} : ret = \text{done } \overline{spine\_elem_i^i} \text{ in } texpr \rangle \longrightarrow \langle h; \sigma(texpr) \rangle} \quad \text{OP\_STE\_TE\_LETT\_SUB}$$

$$\frac{\langle h; seq\_expr \rangle \longrightarrow \langle h; texpr_1 : ret \rangle}{\langle h; \text{let } \overline{ret\_pattern_i^i} = seq\_expr \text{ in } texpr_2 \rangle \longrightarrow \langle h; \text{let } \overline{ret\_pattern_i^i} : ret = texpr_1 \text{ in } texpr_2 \rangle} \quad \text{OP\_STE\_TE\_LETT\_LETT}$$

$$\frac{\langle h; texpr_1 \rangle \longrightarrow \langle h'; texpr'_1 \rangle}{\langle h; \text{let } \overline{ret\_pattern_i^i} : ret = texpr_1 \text{ in } texpr_2 \rangle \longrightarrow \langle h'; \text{let } \overline{ret\_pattern_i^i} : ret = texpr'_1 \text{ in } texpr_2 \rangle} \quad \text{OP\_STE\_TE\_LETT\_LETT}$$



$$\frac{}{\langle h; \text{if True then } texpr_1 \text{ else } texpr_2 \rangle \longrightarrow \langle h; texpr_1 \rangle} \quad \text{OP\_STE\_TE\_IF\_TRUE}$$

$$\frac{}{\langle h; \text{if False then } texpr_1 \text{ else } texpr_2 \rangle \longrightarrow \langle h; texpr_2 \rangle} \quad \text{OP\_STE\_TE\_IF\_FALSE}$$

$$\frac{}{\langle h; \text{bound } [int] (is\_texpr) \rangle \longrightarrow \langle h; is\_texpr \rangle} \quad \text{OP\_STE\_TE\_BOUND}$$

$$\boxed{\langle h; mem\_op \rangle \longrightarrow \langle h'; tval \rangle}$$

$$\frac{bool\_value \equiv mem\_int_1 \text{ binop}_{rel} mem\_int_2}{\langle h; mem\_int_1 \text{ binop}_{rel} mem\_int_2 \rangle \longrightarrow \langle h; \text{done } bool\_value \rangle} \quad \text{OP\_MEMOP\_TVAL\_REL\_BINOP}$$

$$\frac{mem\_int \equiv \text{cast\_ptr\_to\_int } mem\_ptr}{\langle h; \text{intFromPtr } (\tau_1, \tau_2, mem\_ptr) \rangle \longrightarrow \langle h; \text{done } mem\_int \rangle} \quad \text{OP\_MEMOP\_TVAL\_INTFROMPTR}$$

$$\frac{mem\_ptr \equiv \text{cast\_ptr\_to\_int } mem\_int}{\langle h; \text{ptrFromInt } (\tau_1, \tau_2, mem\_int) \rangle \longrightarrow \langle h; \text{done } mem\_ptr \rangle} \quad \text{OP\_MEMOP\_TVAL\_PTRFROMINT}$$

$$\frac{bool\_value \equiv \text{aligned } (\tau, mem\_ptr)}{\langle h + \{mem\_ptr \overset{\checkmark}{\mapsto}_{\tau} -\}; \text{ptrValidForDeref } (\tau, mem\_ptr, mem\_ptr \overset{\checkmark}{\mapsto}_{\tau} -) \rangle \longrightarrow \langle h + \{mem\_ptr \overset{\checkmark}{\mapsto}_{\tau} -\}; \text{done } bool\_value, mem\_ptr \overset{\checkmark}{\mapsto}_{\tau} - \rangle} \quad \text{OP\_MEMOP\_TVAL\_PTRVALID}$$

$$\frac{bool\_value \equiv \text{aligned } (\tau, mem\_ptr)}{\langle h; \text{ptrWellAligned } (\tau, mem\_ptr) \rangle \longrightarrow \langle h; \text{done } bool\_value \rangle} \quad \text{OP\_MEMOP\_TVAL\_PTRWELLALIGNED}$$

$$\frac{mem\_ptr' \equiv mem\_ptr +_{\text{ptr}} (mem\_int \times \text{size\_of}(\tau))}{\langle h; \text{ptrArrayShift } (mem\_ptr, \tau, mem\_int) \rangle \longrightarrow \langle h; \text{done } mem\_ptr' \rangle} \quad \text{OP\_MEMOP\_TVAL\_PTRARRAYSHIFT}$$

$$\boxed{\langle h; mem\_action \rangle \longrightarrow \langle h'; tval \rangle}$$

$$\frac{\begin{array}{l} \text{fresh}(mem\_ptr) \\ \text{representable}(\tau*, mem\_ptr) \\ \text{alignedI}(mem\_int, mem\_ptr) \\ pval:\beta_\tau \end{array}}{\langle h; \text{create}(mem\_int, \tau) \rangle \longrightarrow \langle h + \{mem\_ptr \overset{\times}{\mapsto}_\tau pval\}; \text{done } mem\_ptr, pval, mem\_ptr \overset{\times}{\mapsto}_\tau pval \rangle} \quad \text{OP\_ACTION\_TVAL\_CREATE}$$

$$\frac{}{\langle h + \{mem\_ptr \overset{\check}{\mapsto}_\tau pval\}; \text{load}(\tau, mem\_ptr, -, mem\_ptr \overset{\check}{\mapsto}_\tau pval) \rangle \longrightarrow \langle h + \{mem\_ptr \overset{\check}{\mapsto}_\tau pval\}; \text{done } pval, mem\_ptr \overset{\check}{\mapsto}_\tau pval \rangle} \quad \text{OP\_ACTION\_TVAL\_LOAD}$$

$$\frac{}{\langle h + \{mem\_ptr \overset{\check}{\mapsto}_\tau -\}; \text{store}(-, \tau, mem\_ptr, pval, -, mem\_ptr \overset{\check}{\mapsto}_\tau -) \rangle \longrightarrow \langle h + \{mem\_ptr \overset{\check}{\mapsto}_\tau pval\}; \text{done Unit}, mem\_ptr \overset{\check}{\mapsto}_\tau pval \rangle} \quad \text{OP\_ACTION\_TVAL\_STORE}$$

$$\frac{}{\langle h + \{mem\_ptr \mapsto_\tau -\}; \text{kill}(\text{static } \tau, mem\_ptr, mem\_ptr \mapsto_\tau -) \rangle \longrightarrow \langle h; \text{done Unit} \rangle} \quad \text{OP\_ACTION\_TVAL\_KILL\_STATIC}$$

$$\boxed{\langle h; is\_expr \rangle \longrightarrow \langle h'; is\_expr' \rangle}$$

$$\frac{\langle h; mem\_op \rangle \longrightarrow \langle h; tval \rangle}{\langle h; \text{memop}(mem\_op) \rangle \longrightarrow \langle h; tval \rangle} \quad \text{OP\_ISE\_ISE\_MEMOP}$$

$$\frac{\langle h; mem\_action \rangle \longrightarrow \langle h'; tval \rangle}{\langle h; mem\_action \rangle \longrightarrow \langle h'; tval \rangle} \quad \text{OP\_ISE\_ISE\_ACTION}$$

$$\frac{\langle h; mem\_action \rangle \longrightarrow \langle h'; tval \rangle}{\langle h; \text{neg } mem\_action \rangle \longrightarrow \langle h'; tval \rangle} \quad \text{OP\_ISE\_ISE\_NEG\_ACTION}$$

$$\boxed{\langle h; is\_texpr \rangle \longrightarrow \langle h'; texpr \rangle}$$

$$\frac{\overline{ret\_pattern_i = spine\_elem_i}^i \rightsquigarrow \sigma}{\langle h; \text{let strong } \overline{ret\_pattern_i}^i = \text{done } \overline{spine\_elem_i}^i \text{ in } texpr \rangle \longrightarrow \langle h; \sigma(texpr) \rangle} \quad \text{OP\_ISTE\_ISTE\_LETS\_SUB}$$

$$\frac{\langle h; is\_expr \rangle \longrightarrow \langle h'; is\_expr' \rangle}{\langle h; \text{let strong } \overline{ret\_pattern_i}^i = is\_expr \text{ in } texpr \rangle \longrightarrow \langle h'; \text{let strong } \overline{ret\_pattern_i}^i = is\_expr' \text{ in } texpr \rangle} \quad \text{OP\_ISTE\_ISTE\_LETS\_LETS}$$

$$\boxed{\langle h; texpr \rangle \longrightarrow \langle h'; texpr' \rangle}$$

$$\frac{\langle h; seq\_texpr \rangle \longrightarrow \langle h; texpr \rangle}{\langle h; seq\_texpr \rangle \longrightarrow \langle h; texpr \rangle} \quad \text{OP\_TE\_TE\_SEQ}$$

$$\frac{\langle h; is\_texpr \rangle \longrightarrow \langle h'; texpr \rangle}{\langle h; is\_texpr \rangle \longrightarrow \langle h'; texpr \rangle} \quad \text{OP\_TE\_TE\_IS}$$

$$\boxed{\overline{x_i}^i :: arg \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \mid ret}$$

$$\frac{}{::ret \rightsquigarrow \cdot; \cdot; \cdot; \cdot \mid ret} \quad \text{ARG\_ENV\_RET}$$

$$\frac{\overline{x_i}^i :: arg \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \mid ret}{x, \overline{x_i}^i :: \Pi x:\beta. arg \rightsquigarrow \mathcal{C}, x:\beta; \mathcal{L}; \Phi; \mathcal{R} \mid ret} \quad \text{ARG\_ENV\_COMP}$$

$$\frac{\overline{x_i}^i :: arg \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \mid ret}{x, \overline{x_i}^i :: \forall x:\beta. arg \rightsquigarrow \mathcal{C}; \mathcal{L}, x:\beta; \Phi; \mathcal{R} \mid ret} \quad \text{ARG\_ENV\_LOG}$$

$$\frac{\overline{x_i}^i :: arg \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \mid ret}{\overline{x_i}^i :: term \supset arg \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi, term; \mathcal{R} \mid ret} \quad \text{ARG\_ENV\_PHI}$$

$$\frac{\overline{x_i}^i :: \text{arg} \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \mid \text{ret}}{x, \overline{x_i}^i :: \text{res} \multimap \text{arg} \rightsquigarrow \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}, x:\text{res} \mid \text{ret}} \quad \text{ARG\_ENV\_RES}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}$$

$$\frac{}{\cdot; \cdot; \cdot; \cdot \sqsubseteq \cdot; \cdot; \cdot; \cdot} \quad \text{WEAK\_EMPTY}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}, x:\beta; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}', x:\beta; \mathcal{L}'; \Phi'; \mathcal{R}'} \quad \text{WEAK\_CONS\_COMP}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}, x:\beta; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}', x:\beta; \Phi'; \mathcal{R}'} \quad \text{WEAK\_CONS\_LOG}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}; \Phi, \text{term}; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi', \text{term}; \mathcal{R}'} \quad \text{WEAK\_CONS\_PHI}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}, \text{res} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}', \text{res}} \quad \text{WEAK\_CONS\_RES\_ANON}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}, x:\text{res} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}', x:\text{res}} \quad \text{WEAK\_CONS\_RES\_NAMED}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}', x:\beta; \mathcal{L}'; \Phi'; \mathcal{R}'} \quad \text{WEAK\_SKIP\_COMP}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}', x:\beta; \Phi'; \mathcal{R}'} \quad \text{WEAK\_SKIP\_LOG}$$

$$\frac{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \sqsubseteq \mathcal{C}'; \mathcal{L}'; \Phi'; \text{term}; \mathcal{R}'} \text{WEAK\_SKIP\_PHI}$$

$$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}')}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi; \cdot \vdash (\cdot):(\cdot; \cdot; \cdot; \cdot)} \text{TY\_SUBS\_EMPTY}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}') \\ \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (pval/x, \sigma):(\mathcal{C}', x:\beta; \mathcal{L}'; \Phi'; \mathcal{R}')} \text{TY\_SUBS\_CONS\_COMP}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}') \\ \mathcal{C}; \mathcal{L}; \Phi \vdash pval \Rightarrow \beta \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (pval/x, \sigma):(\mathcal{C}'; \mathcal{L}', x:\beta; \Phi'; \mathcal{R}')} \text{TY\_SUBS\_CONS\_LOG}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}') \\ \text{smt}(\Phi \Rightarrow \text{term}) \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \text{term}; \mathcal{R}')} \text{TY\_SUBS\_CONS\_PHI}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}') \\ \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}_1 \vdash \text{res\_term} \Leftarrow \text{res} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}, \mathcal{R}_1 \vdash (\text{res\_term}/x, \sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}', x:\text{res})} \text{TY\_SUBS\_CONS\_RES\_NAMED}$$

$$\frac{\begin{array}{l} \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}') \\ \mathcal{C}; \mathcal{L}; \Phi; \cdot, \overline{\text{res}_i}^i \vdash \text{res\_term} \Leftarrow \text{res} \end{array}}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R}, \cdot, \overline{\text{res}_i}^i \vdash (\sigma):(\mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}', \cdot, \overline{\text{res}_i}^i)} \text{TY\_SUBS\_CONS\_RES\_ANON}$$

Definition rules: 200 good 0 bad  
Definition rule clauses: 446 good 0 bad