OCaml$_{light}$ in Ott

Scott Owens

OCaml$_{light}$ (ESOP '08) is a formal semantics for a substantial subset of the Objective Caml core language, suitable for writing and verifying real programs.

OCaml$_{light}$ key points

- Written in Ott
- Faithful to Objective Caml (very nearly)
- Type soundness proof mechanized in HOL (Coq and Isabelle/HOL definitions generated too)
- Operational semantics validated on test programs
- Small-step operational semantics (131 rules)
- Type system (179 rules, below)

- definitions:
  - variant data types (e.g., `type t = I of int | C of char`),
  - record types (e.g., `type t = {f : int; g : bool}`),
  - parametric type constructors (e.g., `type 'a t = C of 'a`),
  - type abbreviations (e.g., `type 'a t = 'a * int`),
  - mutually recursive combinations of the above (excepting abbreviations),
  - exceptions, and values;
- expressions for type annotations, sequencing, and primitive values (functions, lists, tuples, and records);
- `with` (record update), `if`, `while`, `for`, `assert`, `try`, and `raise` expressions;
- let-based polymorphism with an SML-style value restriction;
- mutually-recursive function definitions via `let rec`;
- pattern matching, with nested patterns, as patterns, and "or" (|)patterns;
- mutable references with `ref`, `!`, and `:=`;
- polymorphic equality (the Objective Caml = operator);
- 31-bit word semantics for `int`s (using an existing HOL library); and
- IEEE-754 semantics for `float`s (using an existing HOL library).

$\boxed{E \vdash \mathbf{ok}}$ **Environment validity**

$$\frac{}{\mathbf{empty} \vdash \mathbf{ok}}\ \textsf{JTEok\_empty}$$

$$\frac{E \vdash \mathbf{ok}}{E, \mathbf{TV} \vdash \mathbf{ok}}\ \textsf{JTEok\_typevar}$$

$$\frac{E \vdash \forall t : \mathbf{Type}}{E, (\mathit{value\_name} : \forall t) \vdash \mathbf{ok}}\ \textsf{JTEok\_value\_name}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{typeconstr\_name} : \mathit{kind} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{constr\_name} \notin \mathit{names} \end{array}}{E, (\mathit{constr\_name}\ \mathbf{of}\ \mathit{typeconstr\_name}) \vdash \mathbf{ok}}\ \textsf{JTEok\_constr\_name\_c}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{constr\_name} \notin \mathit{names} \end{array}}{E, (\mathit{constr\_name}\ \mathbf{of}\ \mathbf{exn}) \vdash \mathbf{ok}}\ \textsf{JTEok\_exn\_constr\_name\_c}$$

$$\frac{\begin{array}{l} \mathit{type\_params\_opt} = (\alpha_1, \ldots, \alpha_m) \\ E \vdash \forall \mathit{type\_params\_opt}, t_1 : \mathbf{Type} \quad \ldots \quad E \vdash \forall \mathit{type\_params\_opt}, t_n : \mathbf{Type} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{typeconstr\_name} : \mathbf{Type}^m \to \mathbf{Type} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{constr\_name} \notin \mathit{names} \\ \mathbf{length}(t_1)\ldots(t_n) \geq 1 \\ \mathbf{length}(\alpha_1)\ldots(\alpha_m) = m \end{array}}{E, (\mathit{constr\_name}\ \mathbf{of}\ \forall(\alpha_1, \ldots, \alpha_m), (t_1, \ldots, t_n) : \mathit{typeconstr\_name}) \vdash \mathbf{ok}}\ \textsf{JTEok\_constr\_name\_p}$$

$$\frac{\begin{array}{l} E \vdash t_1 : \mathbf{Type} \quad \ldots \quad E \vdash t_n : \mathbf{Type} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{constr\_name} \notin \mathit{names} \\ \mathbf{length}(t_1)\ldots(t_n) \geq 1 \end{array}}{E, (\mathit{constr\_name}\ \mathbf{of}\ \forall, (t_1, \ldots, t_n) : \mathbf{exn}) \vdash \mathbf{ok}}\ \textsf{JTEok\_exn\_constr\_name\_p}$$

$$\frac{\begin{array}{l} E \vdash \forall(\alpha_1, \ldots, \alpha_m), t : \mathbf{Type} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{field\_name} \notin \mathit{names} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{typeconstr\_name} : \mathbf{Type}^m \to \mathbf{Type}\{\mathit{field\_name}_1; \ldots; \mathit{field\_name}_n\} \\ \mathbf{length}(\alpha_1)\ldots(\alpha_m) = m \\ \mathit{field\_name}\ \mathbf{in}\ \mathit{field\_name}_1 \ldots \mathit{field\_name}_n \end{array}}{E, (\mathit{field\_name} : \forall(\alpha_1, \ldots, \alpha_m), \mathit{typeconstr\_name} \to t) \vdash \mathbf{ok}}\ \textsf{JTEok\_record\_destr}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{typeconstr\_name} \notin \mathit{names} \end{array}}{E, (\mathit{typeconstr\_name} : \mathit{kind}) \vdash \mathbf{ok}}\ \textsf{JTEok\_typeconstr\_name}$$

$$\frac{\begin{array}{l} \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{typeconstr\_name} \notin \mathit{names} \\ E \vdash \forall(\alpha_1, \ldots, \alpha_m), t : \mathbf{Type} \end{array}}{E, ((\alpha_1, \ldots, \alpha_m)\ \mathit{typeconstr\_name} = t) \vdash \mathbf{ok}}\ \textsf{JTEok\_typeconstr\_eqn}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{typeconstr\_name} \notin \mathit{names} \\ \mathit{field\_name}_1 \ldots \mathit{field\_name}_n\ \mathbf{distinct} \end{array}}{E, (\mathit{typeconstr\_name} : \mathit{kind}\{\mathit{field\_name}_1; \ldots; \mathit{field\_name}_n\}) \vdash \mathbf{ok}}\ \textsf{JTEok\_typeconstr\_record}$$

$$\frac{\begin{array}{l} E \vdash t : \mathbf{Type} \\ \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{location} \notin \mathit{names} \end{array}}{E, (\mathit{location} : t) \vdash \mathbf{ok}}\ \textsf{JTEok\_location}$$

$\boxed{E \vdash \mathit{typeconstr} : \mathit{kind}}$ **Type constructor kinding**

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{typeconstr\_name} : \mathit{kind} \end{array}}{E \vdash \mathit{typeconstr\_name} : \mathit{kind}}\ \textsf{JTtypeconstr\_abstract}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} = t \\ \vdash \mathit{type\_params\_opt} : \mathit{kind} \end{array}}{E \vdash \mathit{typeconstr\_name} : \mathit{kind}}\ \textsf{JTtypeconstr\_concrete}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{typeconstr\_name} : \mathit{kind}\{\mathit{field\_name}_1; \ldots; \mathit{field\_name}_n\} \end{array}}{E \vdash \mathit{typeconstr\_name} : \mathit{kind}}\ \textsf{JTtypeconstr\_record}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{int} : \mathbf{Type}}\ \textsf{JTtypeconstr\_int}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{char} : \mathbf{Type}}\ \textsf{JTtypeconstr\_char}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{string} : \mathbf{Type}}\ \textsf{JTtypeconstr\_string}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{float} : \mathbf{Type}}\ \textsf{JTtypeconstr\_float}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{bool} : \mathbf{Type}}\ \textsf{JTtypeconstr\_bool}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{unit} : \mathbf{Type}}\ \textsf{JTtypeconstr\_unit}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{exn} : \mathbf{Type}}\ \textsf{JTtypeconstr\_exn}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{list} : \mathbf{Type}^1 \to \mathbf{Type}}\ \textsf{JTtypeconstr\_list}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{option} : \mathbf{Type}^1 \to \mathbf{Type}}\ \textsf{JTtypeconstr\_option}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{ref} : \mathbf{Type}^1 \to \mathbf{Type}}\ \textsf{JTtypeconstr\_ref}$$

$\boxed{E \vdash \mathit{typescheme} : \mathit{kind}}$ **de Bruijn type scheme well-formedness**

$$\frac{E, \mathbf{TV} \vdash t : \mathbf{Type}}{E \vdash \forall t : \mathbf{Type}}\ \textsf{JTts\_forall}$$

$\boxed{E \vdash \forall \mathit{type\_params\_opt}, t : \mathit{kind}}$ **Named type scheme well-formedness**

$$\frac{\begin{array}{l} E \vdash \{\!\{\alpha_1 \leftarrow \mathbf{unit}, \ldots, \alpha_n \leftarrow \mathbf{unit}\}\!\}\ t : \mathbf{Type} \\ \alpha_1 \ldots \alpha_n\ \mathbf{distinct} \end{array}}{E \vdash \forall(\alpha_1, \ldots, \alpha_n), t : \mathbf{Type}}\ \textsf{JTtsnamed\_forall}$$

$\boxed{E \vdash \mathit{typexpr} : \mathit{kind}}$ **Type expression well-formedness**

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{idx}\ \mathbf{bound} \end{array}}{E \vdash <\mathit{idx}, \mathit{num}> : \mathbf{Type}}\ \textsf{JTt\_var}$$

$$\frac{\begin{array}{l} E \vdash t : \mathbf{Type} \\ E \vdash t' : \mathbf{Type} \end{array}}{E \vdash t \to t' : \mathbf{Type}}\ \textsf{JTt\_arrow}$$

$$\frac{\begin{array}{l} E \vdash t_1 : \mathbf{Type} \quad \ldots. \quad E \vdash t_n : \mathbf{Type} \\ \mathbf{length}(t_1)\ldots.(t_n) \geq 2 \end{array}}{E \vdash t_1 * \ldots. * t_n : \mathbf{Type}}\ \textsf{JTt\_tuple}$$

$$\frac{\begin{array}{l} E \vdash \mathit{typeconstr} : \mathbf{Type}^n \to \mathbf{Type} \\ E \vdash t_1 : \mathbf{Type} \quad \ldots \quad E \vdash t_n : \mathbf{Type} \\ \mathbf{length}(t_1)\ldots(t_n) = n \end{array}}{E \vdash (t_1, \ldots, t_n)\ \mathit{typeconstr} : \mathbf{Type}}\ \textsf{JTt\_constr}$$

$\boxed{E \vdash \mathit{typexpr} \equiv \mathit{typexpr}'}$ **Type equivalence**

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash t \equiv t}\ \textsf{JTeq\_refl}$$

$$\frac{E \vdash t' \equiv t}{E \vdash t \equiv t'}\ \textsf{JTeq\_sym}$$

$$\frac{\begin{array}{l} E \vdash t \equiv t' \\ E \vdash t' \equiv t'' \end{array}}{E \vdash t \equiv t''}\ \textsf{JTeq\_trans}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{typeconstr\_name} \rhd (\alpha_1, \ldots, \alpha_n)\ \mathit{typeconstr\_name} = t \\ E \vdash t_1 : \mathbf{Type} \quad \ldots \quad E \vdash t_n : \mathbf{Type} \end{array}}{E \vdash (t_1, \ldots, t_n)\ \mathit{typeconstr\_name} \equiv \{\!\{\alpha_1 \leftarrow t_1, \ldots, \alpha_n \leftarrow t_n\}\!\}\ t}\ \textsf{JTeq\_expand}$$

$$\frac{\begin{array}{l} E \vdash t_1 \equiv t'_1 \\ E \vdash t_2 \equiv t'_2 \end{array}}{E \vdash t_1 \to t_2 \equiv t'_1 \to t'_2}\ \textsf{JTeq\_arrow}$$

$$\frac{\begin{array}{l} E \vdash t_1 \equiv t'_1 \quad \ldots. \quad E \vdash t_n \equiv t'_n \\ \mathbf{length}(t_1)\ldots.(t_n) \geq 2 \end{array}}{E \vdash t_1 * \ldots. * t_n \equiv t'_1 * \ldots. * t'_n}\ \textsf{JTeq\_tuple}$$

$$\frac{\begin{array}{l} E \vdash \mathit{typeconstr} : \mathbf{Type}^n \to \mathbf{Type} \\ E \vdash t_1 \equiv t'_1 \quad \ldots \quad E \vdash t_n \equiv t'_n \\ \mathbf{length}(t_1)\ldots(t_n) = n \end{array}}{E \vdash (t_1, \ldots, t_n)\ \mathit{typeconstr} \equiv (t'_1, \ldots, t'_n)\ \mathit{typeconstr}}\ \textsf{JTeq\_constr}$$

$\boxed{\{\!\{\mathit{typexpr}_1, .., \mathit{typexpr}_n\}\!\}\ \mathit{typexpr}' \rhd \mathit{typexpr}''}$ **de Bruin type substitution**

$$\frac{}{\{\!\{t_1, .., t_n\}\!\}\ \alpha \rhd \alpha}\ \textsf{JTinxsub\_alpha}$$

$$\frac{\mathbf{length}(t_1)..(t_m) = \mathit{num}}{\{\!\{t_1, .., t_m, t', t''_1, .., t''_n\}\!\} < 0, \mathit{num} > \rhd t'}\ \textsf{JTinxsub\_idx0}$$

$$\frac{\mathbf{length}(t_1)..(t_n) \leq \mathit{num}}{\{\!\{t_1, .., t_n\}\!\} < 0, \mathit{num} > \rhd \mathbf{unit}}\ \textsf{JTinxsub\_idx1}$$

$$\frac{}{\{\!\{t_1, .., t_n\}\!\} < \mathit{idx} + 1, \mathit{num} > \rhd < \mathit{idx}, \mathit{num} >}\ \textsf{JTinxsub\_idx2}$$

$$\frac{}{\{\!\{t_1, .., t_n\}\!\}\ \_ \rhd \_}\ \textsf{JTinxsub\_any}$$

$$\frac{\begin{array}{l} \{\!\{t_1, .., t_n\}\!\}\ t'_1 \rhd t''_1 \\ \{\!\{t_1, .., t_n\}\!\}\ t'_2 \rhd t''_2 \end{array}}{\{\!\{t_1, .., t_n\}\!\}\ (t'_1 \to t'_2) \rhd t''_1 \to t''_2}\ \textsf{JTinxsub\_arrow}$$

$$\frac{\{\!\{t_1, .., t_n\}\!\}\ t'_1 \rhd t''_1 \quad \ldots. \quad \{\!\{t_1, .., t_n\}\!\}\ t'_m \rhd t''_m}{\{\!\{t_1, .., t_n\}\!\}\ (t'_1 * \ldots. * t'_m) \rhd (t''_1 * \ldots. * t''_m)}\ \textsf{JTinxsub\_tuple}$$

$$\frac{\{\!\{t_1, .., t_n\}\!\}\ t'_1 \rhd t''_1 \quad \ldots \quad \{\!\{t_1, .., t_n\}\!\}\ t'_m \rhd t''_m}{\{\!\{t_1, .., t_n\}\!\}\ (t'_1, \ldots, t'_m)\ \mathit{typeconstr} \rhd (t''_1, \ldots, t''_m)\ \mathit{typeconstr}}\ \textsf{JTinxsub\_tc}$$

$\boxed{E \vdash \mathit{typexpr} \leq \mathit{typescheme}}$ **de Bruijn type scheme instantiation**

$$\frac{\begin{array}{l} E \vdash \forall t' : \mathbf{Type} \\ E \vdash t_1 : \mathbf{Type} \quad .. \quad E \vdash t_n : \mathbf{Type} \\ \{\!\{t_1, .., t_n\}\!\}\ t' \rhd t'' \end{array}}{E \vdash t'' \leq \forall t'}\ \textsf{JTinst\_idx}$$

$\boxed{E \vdash \mathit{typexpr} \leq \forall \mathit{type\_params\_opt}, \mathit{typexpr}'}$ **Named type scheme instantiation**

$$\frac{\begin{array}{l} E \vdash \forall(\alpha_1, \ldots, \alpha_n), t : \mathbf{Type} \\ E \vdash t_1 : \mathbf{Type} \quad \ldots \quad E \vdash t_n : \mathbf{Type} \end{array}}{E \vdash \{\!\{\alpha_1 \leftarrow t_1, \ldots, \alpha_n \leftarrow t_n\}\!\}\ t \leq \forall(\alpha_1, \ldots, \alpha_n), t}\ \textsf{JTinst\_named\_named}$$

$\boxed{E \vdash \mathit{typexpr} \leq \mathit{typexpr}'}$ **Wildcard type instantiation**

$$\frac{E \vdash < \mathit{idx}, \mathit{num} > : \mathbf{Type}}{E \vdash < \mathit{idx}, \mathit{num} > \leq < \mathit{idx}, \mathit{num} >}\ \textsf{JTinst\_any\_tyvar}$$

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash t \leq \_}\ \textsf{JTinst\_any\_any}$$

$$\frac{\begin{array}{l} E \vdash t_1 \leq t'_1 \\ E \vdash t_2 \leq t'_2 \end{array}}{E \vdash t_1 \to t_2 \leq t'_1 \to t'_2}\ \textsf{JTinst\_any\_arrow}$$

$$\frac{\begin{array}{l} E \vdash t_1 \leq t'_1 \quad \ldots. \quad E \vdash t_n \leq t'_n \\ \mathbf{length}(t_1)\ldots.(t_n) \geq 2 \end{array}}{E \vdash t_1 * \ldots. * t_n \leq t'_1 * \ldots. * t'_n}\ \textsf{JTinst\_any\_tuple}$$

$$\frac{\begin{array}{l} E \vdash t_1 \leq t'_1 \quad \ldots \quad E \vdash t_n \leq t'_n \\ E \vdash \mathit{typeconstr} : \mathbf{Type}^n \to \mathbf{Type} \\ \mathbf{length}(t_1)\ldots(t_n) = n \end{array}}{E \vdash (t_1, \ldots, t_n)\ \mathit{typeconstr} \leq (t'_1, \ldots, t'_n)\ \mathit{typeconstr}}\ \textsf{JTinst\_any\_ctor}$$

$\boxed{E \vdash \mathit{value\_name} : \mathit{typexpr}}$ **Variable typing**

$$\frac{\begin{array}{l} E \vdash \mathit{value\_name} \rhd \mathit{value\_name} : \mathit{ts} \\ E \vdash t \leq \mathit{ts} \end{array}}{E \vdash \mathit{value\_name} : t}\ \textsf{JTvalue\_name\_value\_name}$$

$\boxed{E \vdash \mathit{field\_name} : \mathit{typexpr} \to \mathit{typexpr}'}$ **Field name typing**

$$\frac{\begin{array}{l} E \vdash \mathit{field\_name} \rhd \mathit{field\_name} : \forall(\alpha_1, \ldots, \alpha_m), \mathit{typeconstr\_name} \to t \\ E \vdash (t'_1, \ldots, t'_m)\ \mathit{typeconstr\_name} \to t'' \leq \forall(\alpha_1, \ldots, \alpha_m), (\alpha_1, \ldots, \alpha_m)\ \mathit{typeconstr\_name} \to t \end{array}}{E \vdash \mathit{field\_name} : (t'_1, \ldots, t'_m)\ \mathit{typeconstr\_name} \to t''}\ \textsf{JTfield\_name}$$

$\boxed{E \vdash \mathit{constr} : \mathit{typexpr}_1 \ldots \mathit{typexpr}_n \to \mathit{typexpr}'}$ **Non-constant constructor typing**

$$\frac{\begin{array}{l} E \vdash \mathit{constr\_name} \rhd \mathit{constr\_name}\ \mathbf{of}\ \forall(\alpha_1, \ldots, \alpha_m), (t_1, \ldots., t_n) : \mathit{typeconstr} \\ E \vdash (t'_1 * \ldots. * t'_n) \to (t''_1, \ldots, t''_m)\ \mathit{typeconstr} \leq \forall(\alpha_1, \ldots, \alpha_m), (t_1 * \ldots. * t_n) \to (\alpha_1, \ldots, \alpha_m)\ \mathit{typeconstr} \end{array}}{E \vdash \mathit{constr\_name} : t'_1 \ldots. t'_n \to (t''_1, \ldots, t''_m)\ \mathit{typeconstr}}\ \textsf{JTconstr\_p\_name}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{Invalid\_argument} : \mathbf{string} \to \mathbf{exn}}\ \textsf{JTconstr\_p\_invarg}$$

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash \mathbf{Some} : t \to (t\ \mathbf{option})}\ \textsf{JTconstr\_p\_some}$$

$\boxed{E \vdash \mathit{constr} : \mathit{typexpr}}$ **Constant constructor typing**

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{constr\_name} \rhd \mathit{constr\_name}\ \mathbf{of}\ \mathit{typeconstr\_name} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{typeconstr\_name} : \mathbf{Type}^n \to \mathbf{Type} \\ E \vdash t_1 : \mathbf{Type} \quad \ldots \quad E \vdash t_n : \mathbf{Type} \\ \mathbf{length}(t_1)\ldots(t_n) = n \end{array}}{E \vdash \mathit{constr\_name} : (t_1, \ldots, t_n)\ \mathit{typeconstr\_name}}\ \textsf{JTconstr\_c\_constr}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{constr\_name} \rhd \mathit{constr\_name}\ \mathbf{of}\ \mathbf{exn} \end{array}}{E \vdash \mathit{constr\_name} : \mathbf{exn}}\ \textsf{JTconstr\_c\_exn\_constr}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{Not\_found} : \mathbf{exn}}\ \textsf{JTconstr\_c\_notfound}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{Assert\_failure} : \mathbf{exn}}\ \textsf{JTconstr\_c\_assert\_fail}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{Match\_failure} : \mathbf{exn}}\ \textsf{JTconstr\_c\_match\_fail}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{Division\_by\_zero} : \mathbf{exn}}\ \textsf{JTconstr\_c\_div\_by\_0}$$

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash \mathbf{None} : t\ \mathbf{option}}\ \textsf{JTconstr\_c\_none}$$

$\boxed{E \vdash \mathit{constant} : \mathit{typexpr}}$ **Constant typing**

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathit{integer\_literal} : \mathbf{int}}\ \textsf{JTconst\_int}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathit{float\_literal} : \mathbf{float}}\ \textsf{JTconst\_float}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathit{char\_literal} : \mathbf{char}}\ \textsf{JTconst\_char}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathit{string\_literal} : \mathbf{string}}\ \textsf{JTconst\_string}$$

$$\frac{E \vdash \mathit{constr} : t}{E \vdash \mathit{constr} : t}\ \textsf{JTconst\_constr}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{false} : \mathbf{bool}}\ \textsf{JTconst\_false}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{true} : \mathbf{bool}}\ \textsf{JTconst\_true}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash () : \mathbf{unit}}\ \textsf{JTconst\_unit}$$

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash [] : t\ \mathbf{list}}\ \textsf{JTconst\_nil}$$

$\boxed{\sigma^T \& E \vdash \mathit{pattern} : \mathit{typexpr} \rhd E'}$ **Pattern typing and binding collection**

$$\frac{E \vdash t : \mathbf{Type}}{\sigma^T \& E \vdash x : t \rhd x : t}\ \textsf{JTpat\_var}$$

$$\frac{E \vdash t : \mathbf{Type}}{\sigma^T \& E \vdash \_ : t \rhd \mathbf{empty}}\ \textsf{JTpat\_any}$$

$$\frac{E \vdash \mathit{constant} : t}{\sigma^T \& E \vdash \mathit{constant} : t \rhd \mathbf{empty}}\ \textsf{JTpat\_constant}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern} : t \rhd E' \\ \mathbf{dom}(E', x : t) \rhd \mathit{name}_1 .. \mathit{name}_n \\ \mathit{name}_1 .. \mathit{name}_n\ \mathbf{distinct} \end{array}}{\sigma^T \& E \vdash \mathit{pattern}\ \mathbf{as}\ x : t \rhd E', x : t}\ \textsf{JTpat\_alias}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern} : t \rhd E' \\ E \vdash t' \leq \sigma^T\ \mathit{src\_t} \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash (\mathit{pattern} : \mathit{src\_t}) : t \rhd E'}\ \textsf{JTpat\_typed}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern} : t \rhd E' \\ \sigma^T \& E \vdash \mathit{pattern}' : t \rhd E'' \\ E'\ \mathbf{PERMUTES}\ E'' \end{array}}{\sigma^T \& E \vdash \mathit{pattern} \mid \mathit{pattern}' : t \rhd E'}\ \textsf{JTpat\_or}$$

$$\frac{\begin{array}{l} E \vdash \mathit{constr} : t_1 \ldots t_n \to t \\ \sigma^T \& E \vdash \mathit{pattern}_1 : t_1 \rhd E_1 \quad \ldots \quad \sigma^T \& E \vdash \mathit{pattern}_n : t_n \rhd E_n \\ \mathbf{dom}(E_1 @ \ldots @ E_n) \rhd \mathit{name}_1 .. \mathit{name}_m \\ \mathit{name}_1 .. \mathit{name}_m\ \mathbf{distinct} \end{array}}{\sigma^T \& E \vdash \mathit{constr}(\mathit{pattern}_1, \ldots, \mathit{pattern}_n) : t \rhd E_1 @ \ldots @ E_n}\ \textsf{JTpat\_construct}$$

$$\frac{E \vdash \mathit{constr} : t_1 \ldots t_n \to t}{\sigma^T \& E \vdash \mathit{constr}\ \_ : t \rhd \mathbf{empty}}\ \textsf{JTpat\_construct\_any}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern}_1 : t_1 \rhd E_1 \quad \ldots. \quad \sigma^T \& E \vdash \mathit{pattern}_n : t_n \rhd E_n \\ \mathbf{length}(\mathit{pattern}_1)\ldots.(\mathit{pattern}_n) \geq 2 \\ \mathbf{dom}(E_1 @ \ldots. @ E_n) \rhd \mathit{name}_1 .. \mathit{name}_m \\ \mathit{name}_1 .. \mathit{name}_m\ \mathbf{distinct} \end{array}}{\sigma^T \& E \vdash \mathit{pattern}_1, \ldots., \mathit{pattern}_n : t_1 * \ldots. * t_n \rhd E_1 @ \ldots. @ E_n}\ \textsf{JTpat\_tuple}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern}_1 : t_1 \rhd E_1 \quad \ldots \quad \sigma^T \& E \vdash \mathit{pattern}_n : t_n \rhd E_n \\ E \vdash \mathit{field\_name}_1 : t \to t_1 \quad \ldots \quad E \vdash \mathit{field\_name}_n : t \to t_n \\ \mathbf{length}(\mathit{pattern}_1)\ldots(\mathit{pattern}_n) \geq 1 \\ \mathbf{dom}(E_1 @ \ldots @ E_n) \rhd \mathit{name}_1 .. \mathit{name}_m \\ \mathit{name}_1 .. \mathit{name}_m\ \mathbf{distinct} \end{array}}{\sigma^T \& E \vdash \{\mathit{field\_name}_1 = \mathit{pattern}_1; \ldots; \mathit{field\_name}_n = \mathit{pattern}_n\} : t \rhd E_1 @ \ldots @ E_n}\ \textsf{JTpat\_record}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern} : t \rhd E' \\ \sigma^T \& E \vdash \mathit{pattern}' : t\ \mathbf{list} \rhd E'' \\ \mathbf{dom}(E') \rhd \mathit{name}_1 .. \mathit{name}_m \\ \mathbf{dom}(E'') \rhd \mathit{name}'_1 .. \mathit{name}'_n \\ \mathit{name}_1 .. \mathit{name}_m\ \mathit{name}'_1 .. \mathit{name}'_n\ \mathbf{distinct} \end{array}}{\sigma^T \& E \vdash \mathit{pattern} :: \mathit{pattern}' : t\ \mathbf{list} \rhd E' @ E''}\ \textsf{JTpat\_cons}$$

$\boxed{E \vdash \mathit{unary\_prim} : \mathit{typexpr}}$ **Unary primitive typing**

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash \mathbf{raise} : \mathbf{exn} \to t}\ \textsf{JTuprim\_raise}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \mathbf{not} : \mathbf{bool} \to \mathbf{bool}}\ \textsf{JTuprim\_not}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash \sim- : \mathbf{int} \to \mathbf{int}}\ \textsf{JTuprim\_uminus}$$

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash \mathbf{ref} : t \to (t\ \mathbf{ref})}\ \textsf{JTuprim\_ref}$$

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash\, ! : (t\ \mathbf{ref}) \to t}\ \textsf{JTuprim\_deref}$$

$\boxed{E \vdash \mathit{binary\_prim} : \mathit{typexpr}}$ **Binary primitive typing**

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash\, = : t \to (t \to \mathbf{bool})}\ \textsf{JTbprim\_equal}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash + : \mathbf{int} \to (\mathbf{int} \to \mathbf{int})}\ \textsf{JTbprim\_plus}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash - : \mathbf{int} \to (\mathbf{int} \to \mathbf{int})}\ \textsf{JTbprim\_minus}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash * : \mathbf{int} \to (\mathbf{int} \to \mathbf{int})}\ \textsf{JTbprim\_times}$$

$$\frac{E \vdash \mathbf{ok}}{E \vdash / : \mathbf{int} \to (\mathbf{int} \to \mathbf{int})}\ \textsf{JTbprim\_div}$$

$$\frac{E \vdash t : \mathbf{Type}}{E \vdash := : t\ \mathbf{ref} \to (t \to \mathbf{unit})}\ \textsf{JTbprim\_assign}$$

$\boxed{\sigma^T \& E \vdash \mathit{expr} : \mathit{typexpr}}$ **Expression typing**

$$\frac{\begin{array}{l} E \vdash \mathit{unary\_prim} : t \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash (\%\mathbf{prim}\ \mathit{unary\_prim}) : t'}\ \textsf{JTe\_uprim}$$

$$\frac{\begin{array}{l} E \vdash \mathit{binary\_prim} : t \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash (\%\mathbf{prim}\ \mathit{binary\_prim}) : t'}\ \textsf{JTe\_bprim}$$

$$\frac{\begin{array}{l} E \vdash \mathit{value\_name} : t \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash \mathit{value\_name} : t'}\ \textsf{JTe\_ident}$$

$$\frac{\begin{array}{l} E \vdash \mathit{constant} : t \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash \mathit{constant} : t'}\ \textsf{JTe\_constant}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e : t \\ E \vdash t' \leq \sigma^T\ \mathit{src\_t} \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash (e : \mathit{src\_t}) : t}\ \textsf{JTe\_typed}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : t_1 \quad \ldots. \quad \sigma^T \& E \vdash e_n : t_n \\ \mathbf{length}(e_1)\ldots.(e_n) \geq 2 \\ E \vdash t_1 * \ldots. * t_n \equiv t' \end{array}}{\sigma^T \& E \vdash e_1, \ldots., e_n : t'}\ \textsf{JTe\_tuple}$$

$$\frac{\begin{array}{l} E \vdash \mathit{constr} : t_1 \ldots t_n \to t \\ \sigma^T \& E \vdash e_1 : t_1 \quad \ldots \quad \sigma^T \& E \vdash e_n : t_n \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash \mathit{constr}(e_1, \ldots, e_n) : t'}\ \textsf{JTe\_construct}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : t \\ \sigma^T \& E \vdash e_2 : t\ \mathbf{list} \\ E \vdash t\ \mathbf{list} \equiv t' \end{array}}{\sigma^T \& E \vdash e_1 :: e_2 : t'}\ \textsf{JTe\_cons}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : t_1 \quad \ldots \quad \sigma^T \& E \vdash e_n : t_n \\ E \vdash \mathit{field\_name}_1 : t \to t_1 \quad \ldots \quad E \vdash \mathit{field\_name}_n : t \to t_n \\ t = (t'_1, \ldots, t'_l)\ \mathit{typeconstr\_name} \\ E \vdash \mathit{typeconstr\_name} \rhd \mathit{typeconstr\_name} : \mathit{kind}\{\mathit{field\_name}'_1; \ldots; \mathit{field\_name}'_m\} \\ \mathit{field\_name}_1 \ldots \mathit{field\_name}_n\ \mathbf{PERMUTES}\ \mathit{field\_name}'_1 \ldots \mathit{field\_name}'_m \\ \mathbf{length}(e_1)\ldots(e_n) \geq 1 \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash \{\mathit{field\_name}_1 = e_1; \ldots; \mathit{field\_name}_n = e_n\} : t'}\ \textsf{JTe\_record\_constr}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{expr} : t \\ E \vdash \mathit{field\_name}_1 : t \to t_1 \quad \ldots \quad E \vdash \mathit{field\_name}_n : t \to t_n \\ \sigma^T \& E \vdash e_1 : t_1 \quad \ldots \quad \sigma^T \& E \vdash e_n : t_n \\ \mathit{field\_name}_1 \ldots \mathit{field\_name}_n\ \mathbf{distinct} \\ \mathbf{length}(e_1)\ldots(e_n) \geq 1 \\ E \vdash t \equiv t' \end{array}}{\sigma^T \& E \vdash \{\mathit{expr}\ \mathbf{with}\ \mathit{field\_name}_1 = e_1; \ldots; \mathit{field\_name}_n = e_n\} : t'}\ \textsf{JTe\_record\_with}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e : t_1 \to t \\ \sigma^T \& E \vdash e_1 : t_1 \end{array}}{\sigma^T \& E \vdash e\ e_1 : t}\ \textsf{JTe\_apply}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e : t \\ E \vdash \mathit{field\_name} : t \to t' \\ E \vdash t' \equiv t'' \end{array}}{\sigma^T \& E \vdash e.\mathit{field\_name} : t''}\ \textsf{JTe\_record\_proj}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : \mathbf{bool} \\ \sigma^T \& E \vdash e_2 : \mathbf{bool} \\ E \vdash \mathbf{bool} \equiv t \end{array}}{\sigma^T \& E \vdash e_1 \&\& e_2 : t}\ \textsf{JTe\_and}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : \mathbf{bool} \\ \sigma^T \& E \vdash e_2 : \mathbf{bool} \\ E \vdash \mathbf{bool} \equiv t \end{array}}{\sigma^T \& E \vdash e_1 \,||\, e_2 : t}\ \textsf{JTe\_or}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : \mathbf{bool} \\ \sigma^T \& E \vdash e_2 : t \\ \sigma^T \& E \vdash e_3 : t \end{array}}{\sigma^T \& E \vdash \mathbf{if}\ e_1\ \mathbf{then}\ e_2\ \mathbf{else}\ e_3 : t}\ \textsf{JTe\_ifthenelse}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : \mathbf{bool} \\ \sigma^T \& E \vdash e_2 : \mathbf{unit} \\ E \vdash \mathbf{unit} \equiv t \end{array}}{\sigma^T \& E \vdash \mathbf{while}\ e_1\ \mathbf{do}\ e_2\ \mathbf{done} : t}\ \textsf{JTe\_while}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : \mathbf{int} \\ \sigma^T \& E \vdash e_2 : \mathbf{int} \\ \sigma^T \& E, \mathit{lowercase\_ident} : \mathbf{int} \vdash e_3 : \mathbf{unit} \\ E \vdash \mathbf{unit} \equiv t \end{array}}{\sigma^T \& E \vdash \mathbf{for}\ \mathit{lowercase\_ident} = e_1\ [\mathbf{down}]\mathbf{to}\ e_2\ \mathbf{do}\ e_3\ \mathbf{done} : t}\ \textsf{JTe\_for}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e_1 : \mathbf{unit} \\ \sigma^T \& E \vdash e_2 : t \end{array}}{\sigma^T \& E \vdash e_1; e_2 : t}\ \textsf{JTe\_sequence}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e : t \\ \sigma^T \& E \vdash \mathit{pattern\_matching} : t \to t' \end{array}}{\sigma^T \& E \vdash \mathbf{match}\ e\ \mathbf{with}\ \mathit{pattern\_matching} : t'}\ \textsf{JTe\_match}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern\_matching} : t \to t' \\ E \vdash t \to t' \equiv t'' \end{array}}{\sigma^T \& E \vdash \mathbf{function}\ \mathit{pattern\_matching} : t''}\ \textsf{JTe\_function}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e : t \\ \sigma^T \& E \vdash \mathit{pattern\_matching} : \mathbf{exn} \to t \end{array}}{\sigma^T \& E \vdash \mathbf{try}\ e\ \mathbf{with}\ \mathit{pattern\_matching} : t}\ \textsf{JTe\_try}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pat} = \mathit{expr} \rhd x_1 : t_1, .., x_n : t_n \\ \sigma^T \& E @ x_1 : t_1, .., x_n : t_n \vdash e : t \end{array}}{\sigma^T \& E \vdash \mathbf{let}\ \mathit{pat} = \mathit{expr}\ \mathbf{in}\ e : t}\ \textsf{JTe\_let\_mono}$$

$$\frac{\begin{array}{l} \mathbf{shift}\ 0\ 1\ \sigma^T \& E, \mathbf{TV} \vdash \mathit{pat} = \mathit{nexp} \rhd x_1 : t_1, .., x_n : t_n \\ \sigma^T \& E @ x_1 : \forall t_1, .., x_n : \forall t_n \vdash e : t \end{array}}{\sigma^T \& E \vdash \mathbf{let}\ \mathit{pat} = \mathit{nexp}\ \mathbf{in}\ e : t}\ \textsf{JTe\_let\_poly}$$

$$\frac{\begin{array}{l} \mathbf{shift}\ 0\ 1\ \sigma^T \& E, \mathbf{TV} \vdash \mathit{letrec\_bindings} \rhd x_1 : t_1, \ldots, x_n : t_n \\ \sigma^T \& E @ (x_1 : \forall t_1), \ldots, (x_n : \forall t_n) \vdash e : t \end{array}}{\sigma^T \& E \vdash \mathbf{let\ rec}\ \mathit{letrec\_bindings}\ \mathbf{in}\ e : t}\ \textsf{JTe\_letrec}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash e : \mathbf{bool} \\ E \vdash \mathbf{unit} \equiv t \end{array}}{\sigma^T \& E \vdash \mathbf{assert}\ e : t}\ \textsf{JTe\_assert}$$

$$\frac{E \vdash t : \mathbf{Type}}{\sigma^T \& E \vdash \mathbf{assert\ false} : t}\ \textsf{JTe\_assertfalse}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ E \vdash \mathit{location} \rhd \mathit{location} : t \\ E \vdash t\ \mathbf{ref} \equiv t' \end{array}}{\sigma^T \& E \vdash \mathit{location} : t'}\ \textsf{JTe\_location}$$

$\boxed{\sigma^T \& E \vdash \mathit{pattern\_matching} : \mathit{typexpr} \to \mathit{typexpr}'}$ **Pattern matching/expression pair typing**

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern}_1 : t \rhd E_1 \quad \ldots \quad \sigma^T \& E \vdash \mathit{pattern}_n : t \rhd E_n \\ \sigma^T \& E @ E_1 \vdash e_1 : t' \quad \ldots \quad \sigma^T \& E @ E_n \vdash e_n : t' \\ \mathbf{length}(\mathit{pattern}_1)\ldots(\mathit{pattern}_n) \geq 1 \end{array}}{\sigma^T \& E \vdash \mathit{pattern}_1 \to e_1 \mid \ldots \mid \mathit{pattern}_n \to e_n : t \to t'}\ \textsf{JTpat\_matching\_pm}$$

$\boxed{\sigma^T \& E \vdash \mathit{let\_binding} \rhd E'}$ **Let binding typing**

$$\frac{\begin{array}{l} \sigma^T \& E \vdash \mathit{pattern} : t \rhd x_1 : t_1, .., x_n : t_n \\ \sigma^T \& E \vdash \mathit{expr} : t \end{array}}{\sigma^T \& E \vdash \mathit{pattern} = \mathit{expr} \rhd (x_1 : t_1), .., (x_n : t_n)}\ \textsf{JTlet\_binding\_poly}$$

$\boxed{\sigma^T \& E \vdash \mathit{letrec\_bindings} \rhd E'}$ **Recursive let binding typing**

$$\frac{\begin{array}{l} E' = E @ \mathit{value\_name}_1 : t_1 \to t'_1, \ldots, \mathit{value\_name}_n : t_n \to t'_n \\ \sigma^T \& E' \vdash \mathit{pattern\_matching}_1 : t_1 \to t'_1 \quad \ldots \quad \sigma^T \& E' \vdash \mathit{pattern\_matching}_n : t_n \to t'_n \\ \mathit{value\_name}_1 \ldots \mathit{value\_name}_n\ \mathbf{distinct} \end{array}}{\begin{array}{r} \sigma^T \& E \vdash \mathit{value\_name}_1 = \mathbf{function}\ \mathit{pattern\_matching}_1\ \mathbf{and} \ldots \mathbf{and}\ \mathit{value\_name}_n = \mathbf{function}\ \mathit{pattern\_matching}_n \rhd \\ \mathit{value\_name}_1 : t_1 \to t'_1, \ldots, \mathit{value\_name}_n : t_n \to t'_n \end{array}}\ \textsf{JTletrec\_binding\_equal\_function}$$

$\boxed{\mathit{type\_params\_opt}\ \mathit{typeconstr} \vdash \mathit{constr\_decl} \rhd EB}$ **Variant constructor declaration**

$$\frac{}{(\alpha_1, \ldots, \alpha_n)\ \mathit{typeconstr} \vdash \mathit{constr\_name} \rhd \mathit{constr\_name}\ \mathbf{of}\ \mathit{typeconstr}}\ \textsf{JTconstr\_decl\_nullary}$$

$$\frac{}{(\alpha_1, \ldots, \alpha_n)\ \mathit{typeconstr} \vdash \mathit{constr\_name}\ \mathbf{of}\ t_1 * \ldots * t_n \rhd \mathit{constr\_name}\ \mathbf{of}\ \forall(\alpha_1, \ldots, \alpha_n), (t_1, \ldots, t_n) : \mathit{typeconstr}}\ \textsf{JTconstr\_decl\_nary}$$

$\boxed{\mathit{type\_params\_opt}\ \mathit{typeconstr\_name} \vdash \mathit{field\_decl} \rhd EB}$ **Record field declaration**

$$\frac{}{(\alpha_1, \ldots, \alpha_n)\ \mathit{typeconstr\_name} \vdash \mathit{fn} : t \rhd \mathit{fn} : \forall(\alpha_1, \ldots, \alpha_n), \mathit{typeconstr\_name} \to t}\ \textsf{JTfield\_decl\_only}$$

$\boxed{\vdash \mathit{typedef}_1\ \mathbf{and} .. \mathbf{and}\ \mathit{typedef}_n \rhd E'\ \mathbf{and}\ E''\ \mathbf{and}\ E'''}$ **Type definitions collection**

$$\frac{}{\vdash\ \rhd \mathbf{empty\ and\ empty\ and\ empty}}\ \textsf{JTtypedef\_empty}$$

$$\frac{\vdash \overline{\mathit{typedef}_i}^{\,i} \rhd E\ \mathbf{and}\ E'\ \mathbf{and}\ E''}{\vdash \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} = t\ \mathbf{and}\ \overline{\mathit{typedef}_i}^{\,i} \rhd E\ \mathbf{and}\ E', (\mathit{type\_params\_opt}\ \mathit{typeconstr\_name} = t)\ \mathbf{and}\ E''}\ \textsf{JTtypedef\_eq}$$

$$\frac{\begin{array}{l} \vdash \overline{\mathit{typedef}_i}^{\,i} \rhd E\ \mathbf{and}\ E'\ \mathbf{and}\ E'' \\ \vdash \mathit{type\_params\_opt} : \mathit{kind} \\ \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} \vdash \mathit{constr\_decl}_1 \rhd EB_1 \quad \ldots \quad \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} \vdash \mathit{constr\_decl}_n \rhd EB_n \end{array}}{\begin{array}{r} \vdash \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} = \mathit{constr\_decl}_1 \mid \ldots \mid \mathit{constr\_decl}_n\ \mathbf{and}\ \overline{\mathit{typedef}_i}^{\,i} \rhd \\ E, (\mathit{typeconstr\_name} : \mathit{kind})\ \mathbf{and}\ E'\ \mathbf{and}\ E'' @ EB_1, \ldots, EB_n \end{array}}\ \textsf{JTtypedef\_def\_sum}$$

$$\frac{\begin{array}{l} \vdash \overline{\mathit{typedef}_i}^{\,i} \rhd E\ \mathbf{and}\ E'\ \mathbf{and}\ E'' \\ \vdash \mathit{type\_params\_opt} : \mathit{kind} \\ \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} \vdash \mathit{field\_name}_1 : t_1 \rhd EB_1 \quad \ldots \quad \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} \vdash \mathit{field\_name}_n : t_n \rhd EB_n \end{array}}{\begin{array}{r} \vdash \mathit{type\_params\_opt}\ \mathit{typeconstr\_name} = \{\mathit{field\_name}_1 : t_1; \ldots; \mathit{field\_name}_n : t_n\}\ \mathbf{and}\ \overline{\mathit{typedef}_i}^{\,i} \rhd \\ E, (\mathit{typeconstr\_name} : \mathit{kind}\{\mathit{field\_name}_1; \ldots; \mathit{field\_name}_n\})\ \mathbf{and}\ E'\ \mathbf{and}\ E'' @ EB_1, \ldots, EB_n \end{array}}\ \textsf{JTtypedef\_def\_r}$$

$\boxed{E \vdash \mathit{type\_definition} \rhd E'}$ **Type definition well-formedness and binding collection**

$$\frac{\begin{array}{l} \vdash \mathit{typedef}_1\ \mathbf{and} \ldots \mathbf{and}\ \mathit{typedef}_n \rhd E'\ \mathbf{and}\ E''\ \mathbf{and}\ E''' \\ E'''' = E' @ E'' @ E''' \\ E @ E'''' \vdash \mathbf{ok} \end{array}}{E \vdash \mathbf{type}\ \mathit{typedef}_1\ \mathbf{and} \ldots \mathbf{and}\ \mathit{typedef}_n \rhd E''''}\ \textsf{JTtype\_definition\_list}$$

$$\frac{E \vdash \mathbf{type}\ \overline{\mathit{typedef}_i}^{\,i}\ \mathbf{and}\ \mathit{typedef}'\ \mathbf{and}\ \mathit{typedef}\ \mathbf{and}\ \overline{\mathit{typedef}''_j}^{\,j} \rhd E'}{E \vdash \mathbf{type}\ \overline{\mathit{typedef}_i}^{\,i}\ \mathbf{and}\ \mathit{typedef}\ \mathbf{and}\ \mathit{typedef}'\ \mathbf{and}\ \overline{\mathit{typedef}''_j}^{\,j} \rhd E'}\ \textsf{JTtype\_definition\_swap}$$

$\boxed{E \vdash \mathit{definition} : E'}$ **Definition typing**

$$\frac{\sigma^T \& E, \mathbf{TV} \vdash \mathit{pat} = \mathit{nexp} \rhd (x_1 : t'_1), .., (x_k : t'_k)}{E \vdash \mathbf{let}\ \mathit{pat} = \mathit{nexp} : (x_1 : \forall t'_1), .., (x_k : \forall t'_k)}\ \textsf{JTdefinition\_let\_poly}$$

$$\frac{\sigma^T \& E \vdash \mathit{pat} = \mathit{expr} \rhd (x_1 : t'_1), .., (x_k : t'_k)}{E \vdash \mathbf{let}\ \mathit{pat} = \mathit{expr} : (x_1 : t'_1), .., (x_k : t'_k)}\ \textsf{JTdefinition\_let\_mono}$$

$$\frac{\sigma^T \& E, \mathbf{TV} \vdash \mathit{letrec\_bindings} \rhd (x_1 : t'_1), .., (x_k : t'_k)}{E \vdash \mathbf{let\ rec}\ \mathit{letrec\_bindings} : (x_1 : \forall t'_1), .., (x_k : \forall t'_k)}\ \textsf{JTdefinition\_letrec}$$

$$\frac{E \vdash \mathbf{type}\ \mathit{typedef}_1\ \mathbf{and} \ldots \mathbf{and}\ \mathit{typedef}_n \rhd E'}{E \vdash \mathbf{type}\ \mathit{typedef}_1\ \mathbf{and} \ldots \mathbf{and}\ \mathit{typedef}_n : E'}\ \textsf{JTdefinition\_typedef}$$

$$\frac{\begin{array}{l} E \vdash \mathbf{ok} \\ \mathbf{exn} \vdash \mathit{constr\_decl} \rhd EB \end{array}}{E \vdash \mathbf{exception}\ \mathit{constr\_decl} : EB}\ \textsf{JTdefinition\_exndef}$$

$\boxed{E \vdash \mathit{definitions} : E'}$ **Definition sequence typing**

$$\frac{E \vdash \mathbf{ok}}{E \vdash\ :}\ \textsf{JTdefinitions\_empty}$$

$$\frac{\begin{array}{l} E \vdash \mathit{definition} : E' \\ E @ E' \vdash \mathit{definitions}' : E'' \end{array}}{E \vdash \mathit{definition}\ \mathit{definitions}' : E' @ E''}\ \textsf{JTdefinitions\_item}$$

$\boxed{E \vdash \mathit{program} : E'}$ **Program typing**

$$\frac{E \vdash \mathit{definitions} : E'}{E \vdash \mathit{definitions} : E'}\ \textsf{JTprog\_defs}$$

$$\frac{\sigma^T \& E \vdash v : t}{E \vdash (\%\mathbf{prim\ raise})\ v\ :}\ \textsf{JTprog\_raise}$$

$\boxed{E \vdash \mathit{store} : E'}$ **Store typing**

$$\frac{}{E \vdash \mathbf{empty}\ :}\ \textsf{JTstore\_empty}$$

$$\frac{\begin{array}{l} E \vdash \mathit{store} : E' \\ \{\!\{\ \}\!\} \& E \vdash v : t \end{array}}{E \vdash \mathit{store}, l \mapsto v : E', (l : t)}\ \textsf{JTstore\_map}$$

$\boxed{E \vdash \langle \mathit{program}, \mathit{store} \rangle}$ **Top-level typing**

Checks the combination of a program with a store. The store is typed in an environment that includes its bindings, so that it can contain cyclic structures.

$$\frac{\begin{array}{l} E @ E' \vdash \mathit{store} : E' \\ E @ E' \vdash \mathit{program} : E'' \end{array}}{E \vdash \langle \mathit{program}, \mathit{store} \rangle}\ \textsf{JTtop\_defs}$$

$\boxed{\sigma^T \& E \vdash L}$ **Label-to-environment extraction**

$$\frac{}{\sigma^T \& E \vdash}\ \textsf{JTLin\_nil}$$

$$\frac{\begin{array}{l} \mathbf{dom}(E) \rhd \mathit{names} \\ \mathit{location} \notin \mathit{names} \end{array}}{\sigma^T \& E \vdash \mathbf{ref}\ v = \mathit{location}}\ \textsf{JTLin\_alloc}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash v : t \\ E \vdash \mathit{location} \rhd (\mathit{location} : t) \end{array}}{\sigma^T \& E \vdash\, !\mathit{location} = v}\ \textsf{JTLin\_deref}$$

$$\frac{}{\sigma^T \& E \vdash \mathit{location} := v}\ \textsf{JTLin\_assign}$$

$\boxed{\sigma^T \& E \vdash L \rhd E'}$ **Label-to-environment extraction**

$$\frac{}{\sigma^T \& E \vdash\ \rhd}\ \textsf{JTLout\_nil}$$

$$\frac{\sigma^T \& E \vdash v : t}{\sigma^T \& E \vdash \mathbf{ref}\ v = \mathit{location} \rhd (\mathit{location} : t)}\ \textsf{JTLout\_alloc}$$

$$\frac{}{\sigma^T \& E \vdash\, !\mathit{location} = v \rhd}\ \textsf{JTLout\_deref}$$

$$\frac{\begin{array}{l} \sigma^T \& E \vdash v : t \\ E \vdash \mathit{location} \rhd (\mathit{location} : t) \end{array}}{\sigma^T \& E \vdash \mathit{location} := v \rhd}\ \textsf{JTLout\_assign}$$

The OCaml$_{light}$ Operational Semantics (131 rules)

$\boxed{\vdash \mathit{expr}\ \mathbf{matches}\ \mathit{pattern}}$ **Pattern matching**

$$\frac{}{\vdash v\ \mathbf{matches}\ x}$$ JM_matchP_var

$$\frac{}{\vdash v\ \mathbf{matches}\ \_}$$ JM_matchP_any

$$\frac{}{\vdash \mathit{constant}\ \mathbf{matches}\ \mathit{constant}}$$ JM_matchP_constant

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}}{\vdash v\ \mathbf{matches}\ \mathit{pat}\ \mathbf{as}\ x}$$ JM_matchP_alias

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}}{\vdash v\ \mathbf{matches}\ (\ \mathit{pat}\ :\ t\ )}$$ JM_matchP_typed

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}_1}{\vdash v\ \mathbf{matches}\ \mathit{pat}_1 \mid \mathit{pat}_2}$$ JM_matchP_or_left

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}_2}{\vdash v\ \mathbf{matches}\ \mathit{pat}_1 \mid \mathit{pat}_2}$$ JM_matchP_or_right

$$\frac{\vdash v_1\ \mathbf{matches}\ \mathit{pat}_1 \quad \ldots \quad \vdash v_n\ \mathbf{matches}\ \mathit{pat}_n}{\vdash \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_n\ )\ \mathbf{matches}\ \mathit{constr}\ (\ \mathit{pat}_1\ ,\ \ldots,\ \mathit{pat}_n\ )}$$ JM_matchP_construct

$$\frac{}{\vdash \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_n\ )\ \mathbf{matches}\ \mathit{constr}\ \_}$$ JM_matchP_construct_any

$$\frac{\vdash v_1\ \mathbf{matches}\ \mathit{pat}_1 \quad \ldots. \quad \vdash v_n\ \mathbf{matches}\ \mathit{pat}_n}{\vdash (\ v_1\ ,\ \ldots.,\ v_n\ )\ \mathbf{matches}\ (\ \mathit{pat}_1\ ,\ \ldots.,\ \mathit{pat}_n\ )}$$ JM_matchP_tuple

$$\frac{\begin{array}{l} \mathit{field\_name}'_1\ =\ v'_1 \ldots \mathit{field\_name}'_n\ =\ v'_n\ \mathit{fn}_1\ =\ v''_1 \ldots \mathit{fn}_l\ =\ v''_l\ \mathbf{PERMUTES}\ \mathit{field\_name}_1\ =\ v_1 \ldots \mathit{field\_name}_m\ =\ v_m \\ \vdash v'_1\ \mathbf{matches}\ \mathit{pat}_1 \quad \ldots \quad \vdash v'_n\ \mathbf{matches}\ \mathit{pat}_n \\ \mathit{field\_name}_1 \ldots \mathit{field\_name}_m\ \mathbf{distinct} \end{array}}{\vdash \{\ \mathit{field\_name}_1\ =\ v_1\ ;\ \ldots;\ \mathit{field\_name}_m\ =\ v_m\ \}\ \mathbf{matches}\ \{\ \mathit{field\_name}'_1\ =\ \mathit{pat}_1\ ;\ \ldots;\ \mathit{field\_name}'_n\ =\ \mathit{pat}_n\ \}}$$ JM_matchP_record

$$\frac{\begin{array}{c}\vdash v_1\ \mathbf{matches}\ \mathit{pat}_1 \\ \vdash v_2\ \mathbf{matches}\ \mathit{pat}_2\end{array}}{\vdash v_1 :: v_2\ \mathbf{matches}\ \mathit{pat}_1 :: \mathit{pat}_2}$$ JM_matchP_cons

$\boxed{\vdash \mathit{expr}\ \mathbf{matches}\ \mathit{pattern}\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}\ \}\!\}}$ **Pattern matching with substitution creation**

$$\frac{}{\vdash v\ \mathbf{matches}\ x\ \ \triangleright\ \ \{\!\{\ x \leftarrow v\ \}\!\}}$$ JM_match_var

$$\frac{}{\vdash v\ \mathbf{matches}\ \_\ \ \triangleright\ \ \{\!\{\ \}\!\}}$$ JM_match_any

$$\frac{}{\vdash \mathit{constant}\ \mathbf{matches}\ \mathit{constant}\ \ \triangleright\ \ \{\!\{\ \}\!\}}$$ JM_match_constant

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ \}\!\}}{\vdash v\ \mathbf{matches}\ \mathit{pat}\ \mathbf{as}\ x\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ ,\ x \leftarrow v\ \}\!\}}$$ JM_match_alias

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ \}\!\}}{\vdash v\ \mathbf{matches}\ (\ \mathit{pat}\ :\ t\ )\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ \}\!\}}$$ JM_match_typed

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}_1\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ \}\!\}}{\vdash v\ \mathbf{matches}\ \mathit{pat}_1 \mid \mathit{pat}_2\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ \}\!\}}$$ JM_match_or_left

$$\frac{\begin{array}{l}\neg(v\ \mathbf{matches}\ \mathit{pat}_1) \\ \vdash v\ \mathbf{matches}\ \mathit{pat}_2\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ \}\!\}\end{array}}{\vdash v\ \mathbf{matches}\ \mathit{pat}_1 \mid \mathit{pat}_2\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_n \leftarrow v_n\ \}\!\}}$$ JM_match_or_right

$$\frac{\vdash v_1\ \mathbf{matches}\ \mathit{pat}_1\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ \}\!\} \quad \ldots \quad \vdash v_n\ \mathbf{matches}\ \mathit{pat}_n\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_n\ \}\!\}}{\vdash \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_n\ )\ \mathbf{matches}\ \mathit{constr}\ (\ \mathit{pat}_1\ ,\ \ldots,\ \mathit{pat}_n\ )\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ @ \ldots @\ \mathit{substs\_x}_n\ \}\!\}}$$ JM_match_construct

$$\frac{}{\vdash \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_n\ )\ \mathbf{matches}\ \mathit{constr}\ \_\ \ \triangleright\ \ \{\!\{\ \}\!\}}$$ JM_match_construct_any

$$\frac{\vdash v_1\ \mathbf{matches}\ \mathit{pat}_1\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ \}\!\} \quad \ldots. \quad \vdash v_n\ \mathbf{matches}\ \mathit{pat}_n\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_n\ \}\!\}}{\vdash (\ v_1\ ,\ \ldots.,\ v_n\ )\ \mathbf{matches}\ (\ \mathit{pat}_1\ ,\ \ldots.,\ \mathit{pat}_n\ )\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ @ \ldots. @\ \mathit{substs\_x}_n\ \}\!\}}$$ JM_match_tuple

$$\frac{\begin{array}{l} \mathit{field\_name}'_1\ =\ v'_1 \ldots \mathit{field\_name}'_n\ =\ v'_n\ \mathit{fn}_1\ =\ v''_1 \ldots \mathit{fn}_l\ =\ v''_l\ \mathbf{PERMUTES}\ \mathit{field\_name}_1\ =\ v_1 \ldots \mathit{field\_name}_m\ =\ v_m \\ \vdash v'_1\ \mathbf{matches}\ \mathit{pat}_1\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ \}\!\} \quad \ldots \quad \vdash v'_n\ \mathbf{matches}\ \mathit{pat}_n\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_n\ \}\!\} \\ \mathit{field\_name}_1 \ldots \mathit{field\_name}_m\ \mathbf{distinct} \end{array}}{\vdash \{\ \mathit{field\_name}_1\ =\ v_1\ ;\ \ldots;\ \mathit{field\_name}_m\ =\ v_m\ \}\ \mathbf{matches}\ \{\ \mathit{field\_name}'_1\ =\ \mathit{pat}_1\ ;\ \ldots;\ \mathit{field\_name}'_n\ =\ \mathit{pat}_n\ \}\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ @ \ldots @\ \mathit{substs\_x}_n\ \}\!\}}$$ JM_match_record

$$\frac{\begin{array}{c}\vdash v_1\ \mathbf{matches}\ \mathit{pat}_1\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ \}\!\} \\ \vdash v_2\ \mathbf{matches}\ \mathit{pat}_2\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_2\ \}\!\}\end{array}}{\vdash v_1 :: v_2\ \mathbf{matches}\ \mathit{pat}_1 :: \mathit{pat}_2\ \ \triangleright\ \ \{\!\{\ \mathit{substs\_x}_1\ @\ \mathit{substs\_x}_2\ \}\!\}}$$ JM_match_cons

$\boxed{\mathbf{recfun}\ (\ \mathit{letrec\_bindings}\ ,\ \mathit{pattern\_matching}\ )\ \ \triangleright\ \ \mathit{expr}}$ **Recursive function helper**

$$\frac{\mathit{letrec\_bindings} = (x_1\ =\ \mathbf{function}\ \mathit{pattern\_matching}_1\ \mathbf{and} \ldots \mathbf{and}\ x_n\ =\ \mathbf{function}\ \mathit{pattern\_matching}_n)}{\mathbf{recfun}\ (\ \mathit{letrec\_bindings}\ ,\ \mathit{pattern\_matching}\ )\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow \mathbf{let\ rec}\ \mathit{letrec\_bindings}\ \mathbf{in}\ x_1\ ,\ \ldots,\ x_n \leftarrow \mathbf{let\ rec}\ \mathit{letrec\_bindings}\ \mathbf{in}\ x_n\ \}\!\}\ (\ \mathbf{function}\ \mathit{pattern\_matching}\ )}$$ Jrecfun_letrec

$\boxed{\vdash \mathbf{funval}\ (\ e\ )}$ **Function values**

$$\frac{}{\vdash \mathbf{funval}\ (\ (\ \%\mathbf{prim}\ \mathit{unary\_prim}\ )\ )}$$ Jfunval_up

$$\frac{}{\vdash \mathbf{funval}\ (\ (\ \%\mathbf{prim}\ \mathit{binary\_prim}\ )\ )}$$ Jfunval_bp

$$\frac{}{\vdash \mathbf{funval}\ (\ (\ \%\mathbf{prim}\ \mathit{binary\_prim}\ )\ v\ )}$$ Jfunval_bp_app

$$\frac{}{\vdash \mathbf{funval}\ (\ \mathbf{function}\ \mathit{pattern\_matching}\ )}$$ Jfunval_func

$\boxed{\vdash \mathit{unary\_prim}\ \mathit{expr}\ \overset{L}{\longrightarrow}\ \mathit{expr}'}$ **Unary primitive evaluation**

$$\frac{}{\vdash \mathbf{not\ true}\ \longrightarrow\ \mathbf{false}}$$ Juprim_not_true

$$\frac{}{\vdash \mathbf{not\ false}\ \longrightarrow\ \mathbf{true}}$$ Juprim_not_false

$$\frac{}{\vdash \sim\!-\ \dot{n}\ \longrightarrow\ 0\ \dot{-}\ \dot{n}}$$ Juprim_uminus

$$\frac{}{\vdash \mathbf{ref}\ v\ \overset{\mathbf{ref}\ v\,=\,l}{\longrightarrow}\ l}$$ Juprim_ref_alloc

$$\frac{}{\vdash !\,l\ \overset{!\,l\,=\,v}{\longrightarrow}\ v}$$ Juprim_deref

$\boxed{\vdash \mathit{expr}_1\ \mathit{binary\_prim}\ \mathit{expr}_2\ \overset{L}{\longrightarrow}\ \mathit{expr}}$ **Binary primitive evaluation**

$$\frac{\vdash \mathbf{funval}\ (\ v\ )}{\vdash v\ =\ v'\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ (\ \mathbf{Invalid\_argument}\ (\ \mathbf{equal\_error\_string}\ )\ )}$$ Jbprim_equal_fun

$$\frac{}{\vdash \mathit{constant}\ =\ \mathit{constant}\ \longrightarrow\ \mathbf{true}}$$ Jbprim_equal_const_true

$$\frac{\mathit{constant}\ \neq\ \mathit{constant}'}{\vdash \mathit{constant}\ =\ \mathit{constant}'\ \longrightarrow\ \mathbf{false}}$$ Jbprim_equal_const_false

$$\frac{}{\vdash l\ =\ l'\ \longrightarrow\ (\ (\ \%\mathbf{prim}\ =\ )\ (\ (\ \%\mathbf{prim}\ !\ )\ l\ )\ )\ (\ (\ \%\mathbf{prim}\ !\ )\ l'\ )}$$ Jbprim_equal_loc

$$\frac{}{\vdash (\ v_1 :: v_2\ )\ =\ (\ v'_1 :: v'_2\ )\ \longrightarrow\ (\ (\ (\ \%\mathbf{prim}\ =\ )\ v_1\ )\ v'_1\ )\ \&\&\ (\ (\ (\ \%\mathbf{prim}\ =\ )\ v_2\ )\ v'_2\ )}$$ Jbprim_equal_cons

$$\frac{}{\vdash (\ v_1 :: v_2\ )\ =\ []\ \longrightarrow\ \mathbf{false}}$$ Jbprim_equal_cons_nil

$$\frac{}{\vdash []\ =\ (\ v_1 :: v_2\ )\ \longrightarrow\ \mathbf{false}}$$ Jbprim_equal_nil_cons

$$\frac{\mathbf{length}\ (\ v_1\ ) \ldots. (\ v_n\ )\ \geq\ 2}{\vdash (\ v_1\ ,\ \ldots.,\ v_n\ )\ =\ (\ v'_1\ ,\ \ldots.,\ v'_n\ )\ \longrightarrow\ \mathbf{AND}\ (\ (\ (\ \%\mathbf{prim}\ =\ )\ v_1\ )\ v'_1\ \&\& \ldots. \&\&\ (\ (\ \%\mathbf{prim}\ =\ )\ v_n\ )\ v'_n\ )}$$ Jbprim_equal_tuple

$$\frac{}{\vdash (\ \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_n\ )\ )\ =\ (\ \mathit{constr}\ (\ v'_1\ ,\ \ldots,\ v'_n\ )\ )\ \longrightarrow\ \mathbf{AND}\ (\ (\ (\ \%\mathbf{prim}\ =\ )\ v_1\ )\ v'_1\ \&\& \ldots \&\&\ (\ (\ \%\mathbf{prim}\ =\ )\ v_n\ )\ v'_n\ )}$$ Jbprim_equal_constr

$$\frac{\mathit{constr}\ \neq\ \mathit{constr}'}{\vdash \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_m\ )\ =\ \mathit{constr}'\ (\ v'_1\ ,\ \ldots,\ v'_n\ )\ \longrightarrow\ \mathbf{false}}$$ Jbprim_equal_constr_false

$$\frac{}{\vdash \mathit{constr}'\ =\ \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_n\ )\ \longrightarrow\ \mathbf{false}}$$ Jbprim_equal_const_constr_false

$$\frac{}{\vdash \mathit{constr}\ (\ v_1\ ,\ \ldots,\ v_n\ )\ =\ \mathit{constr}'\ \longrightarrow\ \mathbf{false}}$$ Jbprim_equal_constr_const_false

$$\frac{\begin{array}{c} v'\ =\ \{\ \mathit{fn}''_1\ =\ v''_1\ ;\ \ldots;\ \mathit{fn}''_m\ =\ v''_m\ \} \\ \mathit{fn}_1 \ldots \mathit{fn}_n\ \mathbf{PERMUTES}\ \mathit{fn}''_1 \ldots \mathit{fn}''_m \end{array}}{\vdash \{\ \mathit{fn}_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}_n\ =\ v_n\ \}\ =\ v'\ \longrightarrow\ \mathbf{AND}\ (\ (\ (\ \%\mathbf{prim}\ =\ )\ v_1\ )\ (\ v'\,.\,\mathit{fn}_1\ )\ \&\& \ldots \&\&\ (\ (\ \%\mathbf{prim}\ =\ )\ v_n\ )\ (\ v'\,.\,\mathit{fn}_n\ )\ )}$$ Jbprim_equal_rec

$$\frac{}{\vdash \dot{n}_1\ +\ \dot{n}_2\ \longrightarrow\ \dot{n}_1\ \dot{+}\ \dot{n}_2}$$ Jbprim_plus

$$\frac{}{\vdash \dot{n}_1\ -\ \dot{n}_2\ \longrightarrow\ \dot{n}_1\ \dot{-}\ \dot{n}_2}$$ Jbprim_minus

$$\frac{}{\vdash \dot{n}_1\ *\ \dot{n}_2\ \longrightarrow\ \dot{n}_1\ \dot{*}\ \dot{n}_2}$$ Jbprim_times

$$\frac{}{\vdash \dot{n}\ /\ 0\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ \mathbf{Division\_by\_zero}}$$ Jbprim_div0

$$\frac{\dot{n}_2\ \neq\ 0}{\vdash \dot{n}_1\ /\ \dot{n}_2\ \longrightarrow\ \dot{n}_1\ \dot{/}\ \dot{n}_2}$$ Jbprim_div

$$\frac{}{\vdash l := v\ \overset{l\,:=\,v}{\longrightarrow}\ ()}$$ Jbprim_assign

$\boxed{\vdash \mathit{expr}\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ \mathit{pattern\_matching}'}$ **Pattern matching step**

$$\frac{\begin{array}{c}\neg(v\ \mathbf{matches}\ \mathit{pat}) \\ \mathbf{length}\ (\ e_1\ ) \ldots (\ e_n\ )\ \geq\ 1\end{array}}{\vdash v\ \mathbf{with}\ \mathit{pat}\ \rightarrow\ e \mid \mathit{pat}_1\ \rightarrow\ e_1 \mid \ldots \mid \mathit{pat}_n\ \rightarrow\ e_n\ \longrightarrow\ \mathit{pat}_1\ \rightarrow\ e_1 \mid \ldots \mid \mathit{pat}_n\ \rightarrow\ e_n}$$ JRmatching_next

$\boxed{\vdash \mathit{expr}\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ \mathit{expr}'}$ **Pattern matching finished**

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_m \leftarrow v_m\ \}\!\}}{\vdash v\ \mathbf{with}\ \mathit{pat}\ \rightarrow\ e \mid \mathit{pat}_1\ \rightarrow\ e_1 \mid \ldots \mid \mathit{pat}_n\ \rightarrow\ e_n\ \longrightarrow\ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_m \leftarrow v_m\ \}\!\}\ e}$$ JRmatching_found

$$\frac{\neg(v\ \mathbf{matches}\ \mathit{pat})}{\vdash v\ \mathbf{with}\ \mathit{pat}\ \rightarrow\ e\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ \mathbf{Match\_failure}}$$ JRmatching_fail

$\boxed{\vdash \mathit{store}\ \overset{L}{\longrightarrow}\ \mathit{store}'}$ **Store transition**

$$\frac{}{\vdash \mathit{st}\ \longrightarrow\ \mathit{st}}$$ JRstore_empty

$$\frac{\mathit{st}\ (\ l\ )\ \ \triangleright\ \ v}{\vdash \mathit{st}\ \overset{!\,l\,=\,v}{\longrightarrow}\ \mathit{st}}$$ JRstore_lookup

$$\frac{\mathit{st}'\ (\ l\ )\ \mathbf{unallocated}}{\vdash \mathit{st}\ ,\ l \mapsto \mathit{expr}\ ,\ \mathit{st}'\ \overset{l\,:=\,v}{\longrightarrow}\ \mathit{st}\ ,\ l \mapsto \mathbf{remv\_tyvar}\ v\ ,\ \mathit{st}'}$$ JRstore_assign

$$\frac{\mathit{st}\ (\ l\ )\ \mathbf{unallocated}}{\vdash \mathit{st}\ \overset{\mathbf{ref}\ v\,=\,l}{\longrightarrow}\ \mathit{st}\ ,\ l \mapsto \mathbf{remv\_tyvar}\ v}$$ JRstore_alloc

$\boxed{\mathit{store}\ (\ \mathit{location}\ )\ \ \triangleright\ \ \mathit{expr}}$ **Store lookup**

$$\frac{\begin{array}{c}\mathit{st}\ (\ l\ )\ \ \triangleright\ \ e' \\ l\ \neq\ l'\end{array}}{\mathit{st}\ ,\ l' \mapsto e\ (\ l\ )\ \ \triangleright\ \ e'}$$ JSstlookup_rec

$$\frac{}{\mathit{st}\ ,\ l \mapsto e\ (\ l\ )\ \ \triangleright\ \ e}$$ JSstlookup_found

$\boxed{\vdash \mathit{expr}\ \overset{L}{\longrightarrow}\ \mathit{expr}'}$ **Expression evaluation**

$$\frac{\vdash \mathit{unary\_prim}\ v\ \overset{L}{\longrightarrow}\ e}{\vdash (\ \%\mathbf{prim}\ \mathit{unary\_prim}\ )\ v\ \overset{L}{\longrightarrow}\ e}$$ JR_expr_uprim

$$\frac{\vdash v_1\ \mathit{binary\_prim}\ v_2\ \overset{L}{\longrightarrow}\ e}{\vdash (\ (\ \%\mathbf{prim}\ \mathit{binary\_prim}\ )\ v_1\ )\ v_2\ \overset{L}{\longrightarrow}\ e}$$ JR_expr_bprim

$$\frac{}{\vdash (\ e\ :\ t\ )\ \longrightarrow\ e}$$ JR_expr_typed_ctx

$$\frac{\vdash e_0\ \overset{L}{\longrightarrow}\ e'_0}{\vdash e_1\ e_0\ \overset{L}{\longrightarrow}\ e_1\ e'_0}$$ JR_expr_apply_ctx_arg

$$\frac{}{\vdash e\ (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_apply_raise1

$$\frac{\vdash e_1\ \overset{L}{\longrightarrow}\ e'_1}{\vdash e_1\ v_0\ \overset{L}{\longrightarrow}\ e'_1\ v_0}$$ JR_expr_apply_ctx_fun

$$\frac{}{\vdash (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\ v'\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_apply_raise2

$$\frac{}{\vdash (\ \mathbf{function}\ \mathit{pattern\_matching}\ v_0\ )\ \longrightarrow\ \mathbf{match}\ v_0\ \mathbf{with}\ \mathit{pattern\_matching}}$$ JR_expr_apply

$$\frac{\vdash e_0\ \overset{L}{\longrightarrow}\ e'_0}{\vdash \mathbf{let}\ \mathit{pat}\ =\ e_0\ \mathbf{in}\ e\ \overset{L}{\longrightarrow}\ \mathbf{let}\ \mathit{pat}\ =\ e'_0\ \mathbf{in}\ e}$$ JR_expr_let_ctx

$$\frac{}{\vdash \mathbf{let}\ \mathit{pat}\ =\ (\ \%\mathbf{prim\ raise}\ )\ v\ \mathbf{in}\ e\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_let_raise

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_m \leftarrow v_m\ \}\!\}}{\vdash \mathbf{let}\ \mathit{pat}\ =\ v\ \mathbf{in}\ e\ \longrightarrow\ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_m \leftarrow v_m\ \}\!\}\ e}$$ JR_expr_let_subst

$$\frac{\neg(v\ \mathbf{matches}\ \mathit{pat})}{\vdash \mathbf{let}\ \mathit{pat}\ =\ v\ \mathbf{in}\ e\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ \mathbf{Match\_failure}}$$ JR_expr_let_fail

$$\frac{\begin{array}{l}\mathit{letrec\_bindings} = (x_1\ =\ \mathbf{function}\ \mathit{pattern\_matching}_1\ \mathbf{and} \ldots \mathbf{and}\ x_n\ =\ \mathbf{function}\ \mathit{pattern\_matching}_n) \\ \mathbf{recfun}\ (\ \mathit{letrec\_bindings}\ ,\ \mathit{pattern\_matching}_1\ )\ \ \triangleright\ \ e_1 \quad \ldots \quad \mathbf{recfun}\ (\ \mathit{letrec\_bindings}\ ,\ \mathit{pattern\_matching}_n\ )\ \ \triangleright\ \ e_n\end{array}}{\vdash \mathbf{let\ rec}\ \mathit{letrec\_bindings}\ \mathbf{in}\ e\ \longrightarrow\ \{\!\{\ x_1 \leftarrow e_1\ ,\ \ldots,\ x_n \leftarrow e_n\ \}\!\}\ e}$$ JR_expr_letrec

$$\frac{\vdash e_1\ \overset{L}{\longrightarrow}\ e'_1}{\vdash e_1\ ;\ e_2\ \overset{L}{\longrightarrow}\ e'_1\ ;\ e_2}$$ JR_expr_sequence_ctx_left

$$\frac{}{\vdash (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\ ;\ e\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_sequence_raise

$$\frac{}{\vdash v\ ;\ e_2\ \longrightarrow\ e_2}$$ JR_expr_sequence

$$\frac{\vdash e_1\ \overset{L}{\longrightarrow}\ e'_1}{\vdash \mathbf{if}\ e_1\ \mathbf{then}\ e_2\ \mathbf{else}\ e_3\ \overset{L}{\longrightarrow}\ \mathbf{if}\ e'_1\ \mathbf{then}\ e_2\ \mathbf{else}\ e_3}$$ JR_expr_ifthenelse_ctx

$$\frac{}{\vdash \mathbf{if}\ (\ \%\mathbf{prim\ raise}\ )\ v\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_if_raise

$$\frac{}{\vdash \mathbf{if\ true\ then}\ e_2\ \mathbf{else}\ e_3\ \longrightarrow\ e_2}$$ JR_expr_ifthenelse_true

$$\frac{}{\vdash \mathbf{if\ false\ then}\ e_2\ \mathbf{else}\ e_3\ \longrightarrow\ e_3}$$ JR_expr_ifthenelse_false

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash \mathbf{match}\ e\ \mathbf{with}\ \mathit{pattern\_matching}\ \overset{L}{\longrightarrow}\ \mathbf{match}\ e'\ \mathbf{with}\ \mathit{pattern\_matching}}$$ JR_expr_match_ctx

$$\frac{}{\vdash \mathbf{match}\ (\ \%\mathbf{prim\ raise}\ )\ v\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_match_raise

$$\frac{\vdash v\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ \mathit{pattern\_matching}'}{\vdash \mathbf{match}\ v\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ \mathbf{match}\ v\ \mathbf{with}\ \mathit{pattern\_matching}'}$$ JR_expr_match_step

$$\frac{\vdash v\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ e'}{\vdash \mathbf{match}\ v\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ e'}$$ JR_expr_match_success

$$\frac{}{\vdash e_1\ \&\&\ e_2\ \longrightarrow\ \mathbf{if}\ e_1\ \mathbf{then}\ e_2\ \mathbf{else\ false}}$$ JR_expr_and

$$\frac{}{\vdash e_1\ \|\ e_2\ \longrightarrow\ \mathbf{if}\ e_1\ \mathbf{then\ true\ else}\ e_2}$$ JR_expr_or

$$\frac{}{\vdash \mathbf{while}\ e_1\ \mathbf{do}\ e_2\ \mathbf{done}\ \longrightarrow\ \mathbf{if}\ e_1\ \mathbf{then}\ (\ e_2\ ;\ \mathbf{while}\ e_1\ \mathbf{do}\ e_2\ \mathbf{done}\ )}$$ JR_expr_while

$$\frac{\vdash e_1\ \overset{L}{\longrightarrow}\ e'_1}{\vdash \mathbf{for}\ x\ =\ e_1\ [\mathbf{down}]\mathbf{to}\ e_2\ \mathbf{do}\ e_3\ \mathbf{done}\ \overset{L}{\longrightarrow}\ \mathbf{for}\ x\ =\ e'_1\ [\mathbf{down}]\mathbf{to}\ e_2\ \mathbf{do}\ e_3\ \mathbf{done}}$$ JR_expr_for_ctx1

$$\frac{}{\vdash \mathbf{for}\ x\ =\ (\ \%\mathbf{prim\ raise}\ )\ v\ [\mathbf{down}]\mathbf{to}\ e_2\ \mathbf{do}\ e_3\ \mathbf{done}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_for_raise1

$$\frac{\vdash e_2\ \overset{L}{\longrightarrow}\ e'_2}{\vdash \mathbf{for}\ x\ =\ v_1\ [\mathbf{down}]\mathbf{to}\ e_2\ \mathbf{do}\ e_3\ \mathbf{done}\ \overset{L}{\longrightarrow}\ \mathbf{for}\ x\ =\ v_1\ [\mathbf{down}]\mathbf{to}\ e'_2\ \mathbf{do}\ e_3\ \mathbf{done}}$$ JR_expr_for_ctx2

$$\frac{}{\vdash \mathbf{for}\ x\ =\ v\ [\mathbf{down}]\mathbf{to}\ (\ \%\mathbf{prim\ raise}\ )\ v'\ \mathbf{do}\ e_3\ \mathbf{done}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v'}$$ JR_expr_for_raise2

$$\frac{\dot{n}_1\ \dot{\leq}\ \dot{n}_2}{\vdash \mathbf{for}\ x\ =\ \dot{n}_1\ \mathbf{to}\ \dot{n}_2\ \mathbf{do}\ e\ \mathbf{done}\ \longrightarrow\ (\ \mathbf{let}\ x\ =\ \dot{n}_1\ \mathbf{in}\ e\ )\ ;\ \mathbf{for}\ x\ =\ \dot{n}_1\ \dot{+}\ 1\ \mathbf{to}\ \dot{n}_2\ \mathbf{do}\ e\ \mathbf{done}}$$ JR_expr_for_to_do

$$\frac{\dot{n}_1\ \dot{>}\ \dot{n}_2}{\vdash \mathbf{for}\ x\ =\ \dot{n}_1\ \mathbf{to}\ \dot{n}_2\ \mathbf{do}\ e\ \mathbf{done}\ \longrightarrow\ ()}$$ JR_expr_for_to_done

$$\frac{\dot{n}_2\ \dot{\leq}\ \dot{n}_1}{\vdash \mathbf{for}\ x\ =\ \dot{n}_1\ \mathbf{downto}\ \dot{n}_2\ \mathbf{do}\ e\ \mathbf{done}\ \longrightarrow\ (\ \mathbf{let}\ x\ =\ \dot{n}_1\ \mathbf{in}\ e\ )\ ;\ \mathbf{for}\ x\ =\ \dot{n}_1\ \dot{-}\ 1\ \mathbf{downto}\ \dot{n}_2\ \mathbf{do}\ e\ \mathbf{done}}$$ JR_expr_for_downto_do

$$\frac{\dot{n}_2\ \dot{>}\ \dot{n}_1}{\vdash \mathbf{for}\ x\ =\ \dot{n}_1\ \mathbf{downto}\ \dot{n}_2\ \mathbf{do}\ e\ \mathbf{done}\ \longrightarrow\ ()}$$ JR_expr_for_downto_done

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash \mathbf{try}\ e\ \mathbf{with}\ \mathit{pattern\_matching}\ \overset{L}{\longrightarrow}\ \mathbf{try}\ e'\ \mathbf{with}\ \mathit{pattern\_matching}}$$ JR_expr_try_ctx

$$\frac{}{\vdash \mathbf{try}\ v\ \mathbf{with}\ \mathit{pattern\_matching}\ \longrightarrow\ v}$$ JR_expr_try_return

$$\frac{}{\vdash \mathbf{try}\ (\ \%\mathbf{prim\ raise}\ )\ v\ \mathbf{with}\ \mathit{pat\_exp}_1 \mid \ldots \mid \mathit{pat\_exp}_n\ \longrightarrow\ \mathbf{match}\ v\ \mathbf{with}\ \mathit{pat\_exp}_1 \mid \ldots \mid \mathit{pat\_exp}_n \mid \_\ \rightarrow\ (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )}$$ JR_expr_try_catch

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash e_1\ ,\ ..\ ,\ e_m\ ,\ e\ ,\ v_1\ ,\ ..\ ,\ v_n\ \overset{L}{\longrightarrow}\ e_1\ ,\ ..\ ,\ e_m\ ,\ e'\ ,\ v_1\ ,\ ..\ ,\ v_n}$$ JR_expr_tuple_ctx

$$\frac{}{\vdash e_1\ ,\ ..\ ,\ e_m\ ,\ (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\ ,\ v_1\ ,\ ..\ ,\ v_n\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_tuple_raise

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash \mathit{constr}\ (\ e_1\ ,\ ..\ ,\ e_m\ ,\ e\ ,\ v_1\ ,\ ..\ ,\ v_n\ )\ \overset{L}{\longrightarrow}\ \mathit{constr}\ (\ e_1\ ,\ ..\ ,\ e_m\ ,\ e'\ ,\ v_1\ ,\ ..\ ,\ v_n\ )}$$ JR_expr_constr_ctx

$$\frac{}{\vdash \mathit{constr}\ (\ e_1\ ,\ ..\ ,\ e_m\ ,\ (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\ ,\ v_1\ ,\ ..\ ,\ v_n\ )\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_constr_raise

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash e_0 :: e\ \overset{L}{\longrightarrow}\ e_0 :: e'}$$ JR_expr_cons_ctx1

$$\frac{}{\vdash e :: (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_cons_raise1

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash e :: v\ \overset{L}{\longrightarrow}\ e' :: v}$$ JR_expr_cons_ctx2

$$\frac{}{\vdash (\ (\ \%\mathbf{prim\ raise}\ )\ v\ ) :: v'\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_cons_raise2

$$\frac{\vdash \mathit{expr}\ \overset{L}{\longrightarrow}\ \mathit{expr}'}{\begin{array}{l}\vdash \{\ \mathit{fn}_1\ =\ e_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ e_m\ ;\ \mathit{field\_name}\ =\ \mathit{expr}\ ;\ \mathit{fn}'_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v_n\ \}\ \overset{L}{\longrightarrow} \\ \qquad \{\ \mathit{fn}_1\ =\ e_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ e_m\ ;\ \mathit{field\_name}\ =\ \mathit{expr}'\ ;\ \mathit{fn}'_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v_n\ \}\end{array}}$$ JR_expr_record_ctx

$$\frac{}{\vdash \{\ \mathit{fn}_1\ =\ e_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ e_m\ ;\ \mathit{fn}\ =\ (\ \%\mathbf{prim\ raise}\ )\ v\ ;\ \mathit{fn}'_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v_n\ \}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_record_raise

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\begin{array}{l}\vdash \{\ v\ \mathbf{with}\ \mathit{fn}_1\ =\ e_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ e_m\ ;\ \mathit{field\_name}\ =\ e\ ;\ \mathit{fn}'_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v_n\ \}\ \overset{L}{\longrightarrow} \\ \qquad \{\ v\ \mathbf{with}\ \mathit{fn}_1\ =\ e_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ e_m\ ;\ \mathit{field\_name}\ =\ e'\ ;\ \mathit{fn}'_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v_n\ \}\end{array}}$$ JR_expr_record_with_ctx1

$$\frac{}{\vdash \{\ v'\ \mathbf{with}\ \mathit{fn}_1\ =\ e_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ e_m\ ;\ \mathit{fn}\ =\ (\ \%\mathbf{prim\ raise}\ )\ v\ ;\ \mathit{fn}'_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v_n\ \}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_record_with_raise1

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash \{\ e\ \mathbf{with}\ \mathit{field\_name}_1\ =\ e_1\ ;\ \ldots;\ \mathit{field\_name}_n\ =\ e_n\ \}\ \overset{L}{\longrightarrow}\ \{\ e'\ \mathbf{with}\ \mathit{field\_name}_1\ =\ e_1\ ;\ \ldots;\ \mathit{field\_name}_n\ =\ e_n\ \}}$$ JR_expr_record_with_ctx2

$$\frac{}{\vdash \{\ (\ \%\mathbf{prim\ raise}\ )\ v\ \mathbf{with}\ \mathit{field\_name}_1\ =\ e_1\ ;\ \ldots;\ \mathit{field\_name}_n\ =\ e_n\ \}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_record_raise_ctx2

$$\frac{\begin{array}{l}\mathbf{length}\ (\ v''_1\ ) \ldots (\ v''_l\ )\ \geq\ 1 \\ \mathit{field\_name}\ \notin\ \mathit{fn}_1 \ldots \mathit{fn}_m\end{array}}{\begin{array}{l}\vdash \{\ \{\ \mathit{fn}_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ v_m\ ;\ \mathit{field\_name}\ =\ v\ ;\ \mathit{fn}'_1\ =\ v'_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v'_n\ \}\ \mathbf{with}\ \mathit{field\_name}\ =\ v'\ ;\ \mathit{fn}''_1\ =\ v''_1\ ;\ \ldots;\ \mathit{fn}''_l\ =\ v''_l\ \}\ \longrightarrow \\ \qquad \{\ \{\ \mathit{fn}_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ v_m\ ;\ \mathit{field\_name}\ =\ v'\ ;\ \mathit{fn}'_1\ =\ v'_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v'_n\ \}\ \mathbf{with}\ \mathit{fn}''_1\ =\ v''_1\ ;\ \ldots;\ \mathit{fn}''_l\ =\ v''_l\ \}\end{array}}$$ JR_expr_record_with_many

$$\frac{\mathit{field\_name}\ \notin\ \mathit{fn}_1 \ldots \mathit{fn}_m}{\begin{array}{l}\vdash \{\ \{\ \mathit{fn}_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ v_m\ ;\ \mathit{field\_name}\ =\ v\ ;\ \mathit{fn}'_1\ =\ v'_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v'_n\ \}\ \mathbf{with}\ \mathit{field\_name}\ =\ v'\ \}\ \longrightarrow \\ \qquad \{\ \mathit{fn}_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}_m\ =\ v_m\ ;\ \mathit{field\_name}\ =\ v'\ ;\ \mathit{fn}'_1\ =\ v'_1\ ;\ \ldots;\ \mathit{fn}'_n\ =\ v'_n\ \}\end{array}}$$ JR_expr_record_with_1

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash e\,.\,\mathit{field\_name}\ \overset{L}{\longrightarrow}\ e'\,.\,\mathit{field\_name}}$$ JR_expr_record_access_ctx

$$\frac{}{\vdash (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\,.\,\mathit{field\_name}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_record_access_raise

$$\frac{\mathit{field\_name}\ \notin\ \mathit{fn}_1 \ldots \mathit{fn}_n}{\vdash \{\ \mathit{fn}_1\ =\ v_1\ ;\ \ldots;\ \mathit{fn}_n\ =\ v_n\ ;\ \mathit{field\_name}\ =\ v\ ;\ \mathit{fn}'_1\ =\ v'_1\ ;\ \ldots;\ \mathit{fn}'_m\ =\ v'_m\ \}\,.\,\mathit{field\_name}\ \longrightarrow\ v}$$ JR_expr_record_access

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash \mathbf{assert}\ e\ \overset{L}{\longrightarrow}\ \mathbf{assert}\ e'}$$ JR_expr_assert_ctx

$$\frac{}{\vdash \mathbf{assert}\ (\ (\ \%\mathbf{prim\ raise}\ )\ v\ )\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ v}$$ JR_expr_assert_raise

$$\frac{}{\vdash \mathbf{assert\ true}\ \longrightarrow\ ()}$$ JR_expr_assert_true

$$\frac{}{\vdash \mathbf{assert\ false}\ \longrightarrow\ (\ \%\mathbf{prim\ raise}\ )\ \mathbf{Assert\_failure}}$$ JR_expr_assert_false

$\boxed{\vdash \langle \mathit{definitions}, \mathit{program}\rangle\ \overset{L}{\longrightarrow}\ \langle \mathit{definitions}', \mathit{program}'\rangle}$ **Definition sequence evaluation**

$$\frac{\vdash e\ \overset{L}{\longrightarrow}\ e'}{\vdash \langle \mathit{ds\_value}, \mathbf{let}\ \mathit{pat}\ =\ e\ ;;\ \mathit{definitions}\rangle\ \overset{L}{\longrightarrow}\ \langle \mathit{ds\_value}, \mathbf{let}\ \mathit{pat}\ =\ e'\ ;;\ \mathit{definitions}\rangle}$$ Jdefn_let_ctx

$$\frac{}{\vdash \langle \mathit{ds\_value}, \mathbf{let}\ \mathit{pat}\ =\ (\ \%\mathbf{prim\ raise}\ )\ v\ ;;\ \mathit{definitions}\rangle\ \longrightarrow\ \langle \mathit{ds\_value}, (\%\mathbf{prim\ raise})\ v\rangle}$$ Jdefn_let_raise

$$\frac{\vdash v\ \mathbf{matches}\ \mathit{pat}\ \ \triangleright\ \ \{\!\{\ x_1 \leftarrow v_1\ ,\ ..\ ,\ x_m \leftarrow v_m\ \}\!\}}{\vdash \langle \mathit{ds\_value}, \mathbf{let}\ \mathit{pat}\ =\ v\ ;;\ \mathit{definitions}\rangle\ \longrightarrow\ \langle \mathit{ds\_value}, \{\!\{\ x_1 \leftarrow \mathbf{remv\_tyvar}\ v_1\ ,\ ..\ ,\ x_m \leftarrow \mathbf{remv\_tyvar}\ v_m\ \}\!\}\ \mathit{definitions}\rangle}$$ Jdefn_let_match

$$\frac{\neg(v\ \mathbf{matches}\ \mathit{pat})}{\vdash \langle \mathit{ds\_value}, \mathbf{let}\ \mathit{pat}\ =\ v\ ;;\ \mathit{definitions}\rangle\ \longrightarrow\ \langle \mathit{ds\_value}, (\%\mathbf{prim\ raise})\ \mathbf{Match\_failure}\rangle}$$ Jdefn_let_not_match

$$\frac{\begin{array}{l}\mathit{letrec\_bindings} = (x_1\ =\ \mathbf{function}\ \mathit{pattern\_matching}_1\ \mathbf{and} \ldots \mathbf{and}\ x_n\ =\ \mathbf{function}\ \mathit{pattern\_matching}_n) \\ \mathbf{recfun}\ (\ \mathit{letrec\_bindings}\ ,\ \mathit{pattern\_matching}_1\ )\ \ \triangleright\ \ e_1 \quad \ldots \quad \mathbf{recfun}\ (\ \mathit{letrec\_bindings}\ ,\ \mathit{pattern\_matching}_n\ )\ \ \triangleright\ \ e_n\end{array}}{\vdash \langle \mathit{ds\_value}, \mathbf{let\ rec}\ \mathit{letrec\_bindings}\ ;;\ \mathit{definitions}\rangle\ \longrightarrow\ \langle \mathit{ds\_value}, \{\!\{\ x_1 \leftarrow \mathbf{remv\_tyvar}\ e_1\ ,\ \ldots,\ x_n \leftarrow \mathbf{remv\_tyvar}\ e_n\ \}\!\}\ \mathit{definitions}\rangle}$$ Jdefn_letrec

$$\frac{}{\vdash \langle \mathit{ds\_value}, \mathit{type\_definition}\ ;;\ \mathit{definitions}\rangle\ \longrightarrow\ \langle \mathit{ds\_value}\ ;;\ \mathit{type\_definition}, \mathit{definitions}\rangle}$$ Jdefn_type

$$\frac{}{\vdash \langle \mathit{ds\_value}, \mathit{exception\_definition}\ ;;\ \mathit{definitions}\rangle\ \longrightarrow\ \langle \mathit{ds\_value}\ ;;\ \mathit{exception\_definition}, \mathit{definitions}\rangle}$$ Jdefn_exn

$\boxed{\vdash \langle \mathit{definitions}, \mathit{program}, \mathit{store}\rangle\ \longrightarrow\ \langle \mathit{definitions}', \mathit{program}', \mathit{store}'\rangle}$ **Top-level reduction**

$$\frac{\begin{array}{l}\vdash \mathit{store}\ \overset{L}{\longrightarrow}\ \mathit{store}' \\ \vdash \langle \mathit{definitions\_value}, \mathit{program}\rangle\ \overset{L}{\longrightarrow}\ \langle \mathit{definitions}, \mathit{program}'\rangle\end{array}}{\vdash \langle \mathit{definitions\_value}, \mathit{program}, \mathit{store}\rangle\ \longrightarrow\ \langle \mathit{definitions}, \mathit{program}', \mathit{store}'\rangle}$$ JRtop_defs