Higher-ranked Exception Types

Ruud Koot

February 27, 2015

1 The $\lambda^{\cup}$-calculus

Types

$$\begin{array}{rrll} \tau \in \mathbf{Ty} & ::= & \mathcal{P} & \text{(base type)} \\ & \mid & \tau_1 \to \tau_2 & \text{(function type)} \end{array}$$

Terms

$$\begin{array}{rrll} t \in \mathbf{Tm} & ::= & x, y, \ldots & \text{(variable)} \\ & \mid & \lambda x : \tau . t & \text{(abstraction)} \\ & \mid & t_1\ t_2 & \text{(application)} \\ & \mid & \emptyset & \text{(empty)} \\ & \mid & \{c\} & \text{(singleton)} \\ & \mid & t_1 \cup t_2 & \text{(union)} \end{array}$$

Values Values v are terms of the form

$$\lambda x_1 : \tau_1 . \cdots \lambda x_i : \tau_i . \{c_1\} \cup (\cdots \cup (\{c_j\} \cup (x_1\ v_{11} \cdots v_{1m} \cup (\cdots \cup x_k\ v_{k1} \cdots v_{kn}))))$$

Environments

$$\Gamma \in \mathbf{Env} ::= \quad \cdot \quad \mid \quad \Gamma, x : \tau$$

1.1 Typing relation

$$\frac{}{\Gamma, x : \tau \vdash x : \tau}\ \text{[T-Var]} \qquad \frac{\Gamma, x : \tau_1 \vdash t : \tau_2}{\Gamma \vdash \lambda x : \tau_1 . t : \tau_1 \to \tau_2}\ \text{[T-Abs]} \qquad \frac{\Gamma \vdash t_1 : \tau_1 \to \tau_2 \quad \Gamma \vdash t_2 : \tau_1}{\Gamma \vdash t_1\ t_2 : \tau_2}\ \text{[T-App]}$$

$$\frac{}{\Gamma \vdash \emptyset : \mathcal{P}}\ \text{[T-Empty]} \qquad \frac{}{\Gamma \vdash \{c\} : \mathcal{P}}\ \text{[T-Con]} \qquad \frac{\Gamma \vdash t_1 : \tau \quad \Gamma \vdash t_2 : \tau}{\Gamma \vdash t_1 \cup t_2 : \tau}\ \text{[T-Union]}$$

1.2 Semantics

1.3 Reduction relation

Definition 1. *Let $\prec$ be a strict total order on* $\mathbf{Con} \cup \mathbf{Var}$*, with $c \prec x$ for all $c \in \mathbf{Con}$ and $x \in \mathbf{Var}$.*

$$
\begin{aligned}
(\lambda x : \tau.t_1)\ t_2 &\longrightarrow [t_2/x]\, t_1 && (\beta\text{-reduction})\\
(t_1 \cup t_2)\ t_3 &\longrightarrow t_1\ t_3 \cup t_2\ t_3 &&\\
(\lambda x : \tau.t_1) \cup (\lambda x : \tau.t_2) &\longrightarrow \lambda x : \tau.\, (t_1 \cup t_2) && (\text{congruences})\\
x\ t_1 \cdots t_n \cup x'\ t'_1 \cdots t'_n &\longrightarrow x\ \left(t_1 \cup t'_1\right) \cdots \left(t_n \cup t'_n\right) &&\\
(t_1 \cup t_2) \cup t_3 &\longrightarrow t_1 \cup (t_2 \cup t_3) && (\text{associativity})\\
\emptyset \cup t &\longrightarrow t && (\text{unit})\\
t \cup \emptyset &\longrightarrow t &&\\
x \cup x &\longrightarrow x && (\text{idempotence})\\
x \cup (x \cup t) &\longrightarrow x \cup t &&\\
\{c\} \cup \{c\} &\longrightarrow \{c\} &&\\
\{c\} \cup (\{c\} \cup t) &\longrightarrow \{c\} \cup t &&\\
x\ t_1 \cdots t_n \cup \{c\} &\longrightarrow \{c\} \cup x\ t_1 \cdots t_n && (1)\\
x\ t_1 \cdots t_n \cup (\{c\} \cup t) &\longrightarrow \{c\} \cup (x\ t_1 \cdots t_n \cup t) && (2)\\
x\ t_1 \cdots t_n \cup x'\ t'_1 \cdots t'_n &\longrightarrow x'\ t'_1 \cdots t'_n \cup x\ t_1 \cdots t_n \quad \text{if } x' \prec x && (3)\\
x\ t_1 \cdots t_n \cup \left(x'\ t'_1 \cdots t'_n \cup t\right) &\longrightarrow x'\ t'_1 \cdots t'_n \cup (x\ t_1 \cdots t_n \cup t) \quad \text{if } x' \prec x && (4)\\
\{c\} \cup \{c'\} &\longrightarrow \{c'\} \cup \{c\} \quad \text{if } c' \prec c && (5)\\
\{c\} \cup \left(\{c'\} \cup t\right) &\longrightarrow \{c'\} \cup (\{c\} \cup t) \quad \text{if } c' \prec c && (6)
\end{aligned}
$$

Conjecture 1. *The reduction relation $\longrightarrow$ preserves meaning.*

Conjecture 2. *The reduction relation $\longrightarrow$ is strongly normalizing.*

Conjecture 3. *The reduction relation $\longrightarrow$ is locally confluent.*

Corollary 1. *The reduction relation $\longrightarrow$ is confluent.*

Proof. Follows from SN, LC and Newman's Lemma. □

Corollary 2. *The $\lambda^{\cup}$-calculus has unique normal forms.*

Corollary 3. *Equality of $\lambda^{\cup}$-terms can be decided by normalization.*

2 Completion

$$
\begin{aligned}
\kappa \in \mathbf{Kind} \quad &::= \quad \mathrm{E} && (\text{exception})\\
&\mid \quad \kappa_1 \Rightarrow \kappa_2 && (\text{exception operator})
\end{aligned}
$$

$$
\begin{array}{rcll}
\varphi \in \mathbf{Exn} & ::= & e & \text{(exception variables)} \\
& \mid & \lambda e : \kappa.\varphi & \text{(exception abstraction)}
\end{array}
$$

$$
\begin{array}{rcll}
\widehat{\tau} \in \mathbf{ExnTy} & ::= & \forall e :: \kappa.\widehat{\tau} & \text{(exception quantification)} \\
& \mid & \widehat{\mathrm{bool}} & \text{(boolean type)} \\
& \mid & [\widehat{\tau}\langle\varphi\rangle] & \text{(list type)} \\
& \mid & \widehat{\tau}_1\langle\varphi_1\rangle \to \widehat{\tau}_2\langle\varphi_2\rangle & \text{(function type)}
\end{array}
$$

The completion procedure as a set of inference rules:

$$\frac{}{\overline{e_i :: \kappa_i} \vdash \mathbf{bool} : \widehat{\mathrm{bool}} \;\&\; e\ \overline{e_i} \triangleright e :: \overline{\kappa_i \Rightarrow \mathrm{E}}} \text{ [C-Bool]}$$

$$\frac{\overline{e_i :: \kappa_i} \vdash \tau : \widehat{\tau} \;\&\; \varphi \triangleright \overline{e_j :: \kappa_j}}{\overline{e_i :: \kappa_i} \vdash [\tau] : [\widehat{\tau}\langle\varphi\rangle] \;\&\; e\ \overline{e_i} \triangleright e :: \overline{\kappa_i \Rightarrow \mathrm{E}}, \overline{e_j :: \kappa_j}} \text{ [C-List]}$$

$$\frac{\vdash \tau_1 : \widehat{\tau}_1 \;\&\; \varphi_1 \triangleright \overline{e_j :: \kappa_j} \quad \overline{e_i :: \kappa_i}, \overline{e_j :: \kappa_j} \vdash \tau_2 : \widehat{\tau}_2 \;\&\; \varphi_2 \triangleright \overline{e_j :: \kappa_j}}{\overline{e_i :: \kappa_i} \vdash \tau_1 \to \tau_2 : \forall \overline{e_j :: \kappa_j}.\, (\widehat{\tau}_1\langle\varphi_1\rangle \to \widehat{\tau}_2\langle\varphi_2\rangle) \;\&\; e\ \overline{e_i} \triangleright e :: \overline{\kappa_j \Rightarrow \mathrm{E}}, \overline{e_k :: \kappa_k}} \text{ [C-Arr]}$$

Figure 1: Type completion ($\Gamma \vdash \tau : \widehat{\tau} \;\&\; \varphi \triangleright \Gamma'$)

The completion procedure as an algorithm:

complete :: **Env** × **Ty** → **ExnTy** × **Exn** × **Env**
complete $\overline{e_i :: \kappa_i}$ **bool** =
  let *e be fresh*
  in $\langle \widehat{\mathrm{bool}}; e\ \overline{e_i}; e :: \overline{\kappa_i \Rightarrow \mathrm{E}} \rangle$

3 Type system

3.1 Terms

$$
\begin{array}{lcll}
t \in \mathbf{Tm} & ::= & x & \text{(term variable)} \\
& | & c_\tau & \text{(term constant)} \\
& | & \lambda x : \tau . t & \text{(term abstraction)} \\
& | & t_1\ t_2 & \text{(term application)} \\
& | & t_1 \oplus t_2 & \text{(operator)} \\
& | & \textbf{if } t_1 \textbf{ then } t_2 \textbf{ else } t_3 & \text{(conditional)} \\
& | & \lightning^{\ell}_{\tau} & \text{(exception constant)} \\
& | & t_1 \textbf{ seq } t_2 & \text{(forcing)} \\
& | & \textbf{fix } t & \text{(anonymous fixpoint)} \\
& | & []_\tau & \text{(nil constructor)} \\
& | & t_1 :: t_2 & \text{(cons constructor)} \\
& | & \textbf{case } t_1 \textbf{ of } \{[] \mapsto t_2; x_1 :: x_2 \mapsto t_3\} & \text{(list eliminator)}
\end{array}
$$

3.2 Underlying type system

$$\frac{}{\Gamma, x : \tau \vdash x : \tau}\ [\text{T-Var}] \qquad \frac{}{\Gamma \vdash c_\tau : \tau}\ [\text{T-Con}] \qquad \frac{}{\Gamma \vdash \lightning^{\ell}_{\tau} : \tau}\ [\text{T-Crash}]$$

$$\frac{\Gamma, x : \tau_1 \vdash t : \tau_2}{\Gamma \vdash \lambda x : \tau_1 . t : \tau_1 \to \tau_2}\ [\text{T-Abs}] \qquad \frac{\Gamma \vdash t_1 : \tau_2 \to \tau \quad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash t_1\ t_2 : \tau}\ [\text{T-App}]$$

$$\frac{\Gamma \vdash t : \tau \to \tau}{\Gamma \vdash \textbf{fix } t : \tau}\ [\text{T-Fix}]$$

$$\frac{\Gamma \vdash t_1 : \textbf{int} \quad \Gamma \vdash t_2 : \textbf{int}}{\Gamma \vdash t_1 \oplus t_2 : \textbf{bool}}\ [\text{T-Op}] \qquad \frac{\Gamma \vdash t_1 : \tau_1 \quad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash t_1 \textbf{ seq } t_2 : \tau_2}\ [\text{T-Seq}]$$

$$\frac{\Gamma \vdash t_1 : \textbf{bool} \quad \Gamma \vdash t_2 : \tau \quad \Gamma \vdash t_3 : \tau}{\Gamma \vdash \textbf{if } t_1 \textbf{ then } t_2 \textbf{ else } t_3 : \tau}\ [\text{T-If}]$$

$$\frac{}{\Gamma \vdash []_\tau : [\tau]}\ [\text{T-Nil}] \qquad \frac{\Gamma \vdash t_1 : \tau \quad \Gamma \vdash t_2 : [\tau]}{\Gamma \vdash t_1 :: t_2 : [\tau]}\ [\text{T-Cons}]$$

$$\frac{\Gamma \vdash t_1 : [\tau_1] \quad \Gamma \vdash t_2 : \tau \quad \Gamma, x_1 : \tau_1, x_2 : [\tau_1] \vdash t_3 : \tau}{\Gamma \vdash \textbf{case } t_1 \textbf{ of } \{[] \mapsto t_2; x_1 :: x_2 \mapsto t_3\} : \tau}\ [\text{T-Case}]$$

Figure 2: Underlying type system ($\Gamma \vdash t : \tau$)

3.3 Declarative exception type system

- In T-Abs and T-AnnAbs, should the term-level term-abstraction also have an explicit effect annotation?
- In T-AnnAbs, might need a side condition stating that e is not free in Δ.
- In T-App, note the double occurence of φ when typing t_1. Is subeffecting sufficient here? Also note that we do *not* expect an exception variable in the left-hand side annotation of the function space constructor.
- In T-AnnApp, note the substitution. We will need a substitution lemma for annotations.
- In T-Fix, the might be some universal quantifiers in our way. Do annotation applications in t take care of this, already? Perhaps we do need to change **fix** t into a binding construct to resolve this? Also, there is some implicit subeffecting going on between the annotations and effect.
- In T-Case, note the use of explicit subeffecting. Can this be done using implicit subeffecting?
- For T-Sub, should we introduce a term-level coercion, as in Dussart–Henglein–Mossin? We now do shape-conformant subtyping, is subeffecting sufficient?
- Do we need additional kinding judgements in some of the rules? Can we merge the kinding judgement with the subtyping and/or -effecting judgement? Kind-preserving substitutions.

3.4 Type elaboration system

- For T-Fix: how would a binding fixpoint construct work?

3.5 Type inference algorithm

$$
\begin{array}{lll}
\multicolumn{3}{l}{\mathcal{R} : \mathbf{Env} \times \mathbf{KiEnv} \times \mathbf{Tm} \to \mathbf{ExnTy} \times \mathbf{Exn}} \\
\mathcal{R}\ \Gamma\ \Delta\ x & = \Gamma_x & \\
\mathcal{R}\ \Gamma\ \Delta\ c_\tau & = \langle \bot_\tau ; \emptyset \rangle & \\
\mathcal{R}\ \Gamma\ \Delta\ \lightning^\ell_\tau & = \langle \bot_\tau ; \{\ell\} \rangle & \\
\mathcal{R}\ \Gamma\ \Delta\ \lambda x : \tau . t & = \textbf{let}\ \langle \widehat{\tau}_1 ; e_1 ; \overline{e_i : \kappa_i} \rangle = \mathcal{C}\ \emptyset\ \tau & \\
 & \qquad \langle \widehat{\tau}_2 ; \varphi_2 \rangle \quad = \mathcal{R}\ \Gamma, \widehat{\tau}_1\ \&\ e_1\ \Delta, \overline{e_i : \kappa_i}\ t & \\
 & \quad \textbf{in}\ \ \langle \forall \overline{e_i : \kappa_i} . \widehat{\tau}_1 \langle e_1 \rangle \to \widehat{\tau}_2 \langle \varphi_2 \rangle ; \emptyset \rangle & \\
\mathcal{R}\ \Gamma\ \Delta\ (t_1\ t_2) & = \textbf{let}\ \langle \widehat{\tau}_1 ; \varphi_1 \rangle = \mathcal{R}\ \Gamma\ \Delta\ t_1 & \\
 & \qquad \langle \widehat{\tau}_2 ; \varphi_2 \rangle = \mathcal{R}\ \Gamma\ \Delta\ t_2 &
\end{array}
$$

- Note that we do not construct the elaborated term, as it is not useful except for metatheoretic purposes.
- Simplification does not exactly match the prototype (update the latter).

- Lemma: The algorithm maintains the invariant that exception types and exceptions are in normal form.

$$\frac{}{\Gamma, x : \widehat{\tau} \;\&\; \varphi; \Delta \vdash x : \widehat{\tau} \;\&\; \varphi}\ [\text{T-Var}]$$

$$\frac{}{\Gamma; \Delta \vdash c_\tau : \bot_\tau \;\&\; \emptyset}\ [\text{T-Con}] \qquad \frac{}{\Gamma; \Delta \vdash \lightning^{\ell}_{\tau} : \bot_\tau \;\&\; \{\ell\}}\ [\text{T-Crash}]$$

$$\frac{\Gamma, x : \widehat{\tau}_1 \;\&\; \varphi_1; \Delta \vdash t : \widehat{\tau}_2 \;\&\; \varphi_2}{\Gamma; \Delta \vdash \lambda x : \widehat{\tau}_1 \;\&\; \varphi_1 . t : \widehat{\tau}_1\langle\varphi_1\rangle \to \widehat{\tau}_2\langle\varphi_2\rangle \;\&\; \emptyset}\ [\text{T-Abs}]$$

$$\frac{\Gamma; \Delta, e : \kappa \vdash t : \widehat{\tau} \;\&\; \varphi}{\Gamma; \Delta \vdash \Lambda e : \kappa . t : \forall e : \kappa . \widehat{\tau} \;\&\; \varphi}\ [\text{T-AnnAbs}]$$

$$\frac{\Gamma; \Delta \vdash t_1 : \widehat{\tau}_2\langle\varphi_2\rangle \to \widehat{\tau}\langle\varphi\rangle \;\&\; \varphi \quad \Gamma; \Delta \vdash t_2 : \widehat{\tau}_2 \;\&\; \varphi_2}{\Gamma; \Delta \vdash t_1\ t_2 : \widehat{\tau} \;\&\; \varphi}\ [\text{T-App}]$$

$$\frac{\Gamma; \Delta \vdash t_1 : \forall e : \kappa . \widehat{\tau} \;\&\; \varphi \quad \Delta \vdash \varphi_2 : \kappa}{\Gamma; \Delta \vdash t_1\ \langle\varphi_2\rangle : [\varphi_2/e]\,\widehat{\tau} \;\&\; \varphi}\ [\text{T-AnnApp}]$$

$$\frac{\Gamma; \Delta \vdash t : \widehat{\tau}\langle\varphi\rangle \to \widehat{\tau}\langle\varphi\rangle \;\&\; \varphi}{\Gamma; \Delta \vdash \mathbf{fix}\ t : \widehat{\tau} \;\&\; \varphi}\ [\text{T-Fix}]$$

$$\frac{\Gamma; \Delta \vdash t_1 : \widehat{\mathbf{int}} \;\&\; \varphi \quad \Gamma; \Delta \vdash t_2 : \widehat{\mathbf{int}} \;\&\; \varphi}{\Gamma; \Delta \vdash t_1 \oplus t_2 : \widehat{\mathbf{bool}} \;\&\; \varphi}\ [\text{T-Op}]$$

$$\frac{\Gamma; \Delta \vdash t_1 : \widehat{\tau}_1 \;\&\; \varphi \quad \Gamma; \Delta \vdash t_2 : \widehat{\tau}_2 \;\&\; \varphi}{\Gamma; \Delta \vdash t_1\ \mathbf{seq}\ t_2 : \widehat{\tau}_2 \;\&\; \varphi}\ [\text{T-Seq}]$$

$$\frac{\Gamma; \Delta \vdash t_1 : \widehat{\mathbf{bool}} \;\&\; \varphi \quad \Gamma; \Delta \vdash t_2 : \widehat{\tau} \;\&\; \varphi \quad \Gamma; \Delta \vdash t_3 : \widehat{\tau} \;\&\; \varphi}{\Gamma; \Delta \vdash \mathbf{if}\ t_1\ \mathbf{then}\ t_2\ \mathbf{else}\ t_3 : \widehat{\tau} \;\&\; \varphi}\ [\text{T-If}]$$

$$\frac{}{\Gamma; \Delta \vdash []_\tau : [\bot_\tau\langle\emptyset\rangle] \;\&\; \emptyset}\ [\text{T-Nil}]$$

$$\frac{\Gamma; \Delta \vdash t_1 : \widehat{\tau} \;\&\; \varphi_1 \quad \Gamma; \Delta \vdash t_2 : [\widehat{\tau}\langle\varphi_1\rangle] \;\&\; \varphi_2}{\Gamma; \Delta \vdash t_1 :: t_2 : [\widehat{\tau}\langle\varphi_1\rangle] \;\&\; \varphi_2}\ [\text{T-Cons}]$$

$$\frac{\begin{array}{c}\Gamma; \Delta \vdash t_1 : [\widehat{\tau}_1\langle\varphi_1\rangle] \;\&\; \varphi' \quad \Delta \vdash \varphi' \leqslant \varphi \quad \Gamma; \Delta \vdash t_2 : \widehat{\tau} \;\&\; \varphi \\ \Gamma, x_1 : \widehat{\tau}_1 \;\&\; \varphi_1, x_2 : [\widehat{\tau}_1\langle\varphi_1\rangle] \;\&\; \varphi'; \Delta \vdash t_3 : \widehat{\tau} \;\&\; \varphi\end{array}}{\Gamma; \Delta \vdash \mathbf{case}\ t_1\ \mathbf{of}\ \{[] \mapsto t_2; x_1 :: x_2 \mapsto t_3\} : \widehat{\tau} \;\&\; \varphi}\ [\text{T-Case}]$$

$$\frac{\Gamma; \Delta \vdash t : \widehat{\tau} \;\&\; \varphi \quad \Delta \vdash \widehat{\tau} \leqslant \widehat{\tau}' \quad \Delta \vdash \varphi \leqslant \varphi'}{\Gamma; \Delta \vdash t : \widehat{\tau}' \;\&\; \varphi'}\ [\text{T-Sub}]$$

Figure 3: Declarative type system ($\Gamma; \Delta \vdash t : \widehat{\tau} \;\&\; \varphi$)

$$\frac{}{\Gamma, x : \widehat{\tau} \;\&\; \varphi; \Delta \vdash x \rightsquigarrow x : \widehat{\tau} \;\&\; \varphi} \text{ [T-Var]}$$

$$\frac{}{\Gamma; \Delta \vdash c_\tau \rightsquigarrow c_\tau : \tau \;\&\; \emptyset} \text{ [T-Con]} \qquad \frac{}{\Gamma; \Delta \vdash \lightning^{\ell}_{\tau} \rightsquigarrow \lightning^{\ell}_{\tau} : \bot_\tau \;\&\; \{\ell\}} \text{ [T-Crash]}$$

$$\frac{\begin{array}{c}\Delta, \overline{e_i : \kappa_i} \vdash \widehat{\tau}_1 \triangleright \tau_1 \quad \Delta, \overline{e_i : \kappa_i} \vdash \varphi_1 : \mathrm{E} \\ \Gamma, x : \widehat{\tau}_1 \;\&\; \varphi_1; \Delta, \overline{e_i : \kappa_i} \vdash t \rightsquigarrow t' : \widehat{\tau}_2 \;\&\; \varphi_2\end{array}}{\Gamma; \Delta \vdash \lambda x : \tau_1.t \rightsquigarrow \lambda x : \widehat{\tau}_1 \;\&\; \varphi_1.t' : \forall \overline{e_i : \kappa_i}.\widehat{\tau}_1\langle\varphi_1\rangle \to \widehat{\tau}_2\langle\varphi_2\rangle \;\&\; \emptyset} \text{ [T-Abs]}$$

$$\frac{\begin{array}{c}\Delta \vdash \widehat{\tau}_2 \leqslant [\overline{\varphi_i}/\overline{e_i}]\,\widehat{\tau} \quad \Delta \vdash \varphi_2 \leqslant [\overline{\varphi_i}/\overline{e_i}]\,\varphi \quad \overline{\Delta \vdash \varphi_i : \kappa_i} \\ \Gamma; \Delta \vdash t_1 \rightsquigarrow t_1' : \forall \overline{e_i : \kappa_i}.\widehat{\tau}_1\langle\varphi_1\rangle \to \widehat{\tau}\langle\varphi\rangle \;\&\; \varphi' \quad \Gamma; \Delta \vdash t_2 \rightsquigarrow t_2' : \widehat{\tau}_2 \;\&\; \varphi_2\end{array}}{\Gamma; \Delta \vdash t_1\ t_2 \rightsquigarrow t_1'\ \langle\overline{\varphi_i}\rangle\ t_2' : [\overline{\varphi_i}/\overline{e_i}]\,\widehat{\tau} \;\&\; [\overline{\varphi_i}/\overline{e_i}]\,\varphi \cup \varphi'} \text{ [T-App]}$$

$$\frac{\begin{array}{c}\Gamma; \Delta \vdash t \rightsquigarrow t' : \forall \overline{e_i : \kappa_i}.\widehat{\tau}\langle\varphi\rangle \to \widehat{\tau}'\langle\varphi'\rangle \;\&\; \varphi'' \\ \Delta \vdash [\overline{\varphi_i}/\overline{e_i}]\,\widehat{\tau}' \leqslant [\overline{\varphi_i}/\overline{e_i}]\,\widehat{\tau} \quad \Delta \vdash [\overline{\varphi_i}/\overline{e_i}]\,\varphi' \leqslant [\overline{\varphi_i}/\overline{e_i}]\,\varphi \quad \overline{\Delta \vdash \varphi_i : \kappa_i}\end{array}}{\Gamma; \Delta \vdash \mathbf{fix}\ t \rightsquigarrow \mathbf{fix}\ t' : [\overline{\varphi_i}/\overline{e_i}]\,\widehat{\tau} \;\&\; [\overline{\varphi_i}/\overline{e_i}]\,\varphi \cup \varphi''} \text{ [T-Fix]}$$

$$\frac{\Gamma; \Delta \vdash t_1 \rightsquigarrow t_1' : \widehat{\mathbf{int}} \;\&\; \varphi_1 \quad \Gamma; \Delta \vdash t_2 \rightsquigarrow t_2' : \widehat{\mathbf{int}} \;\&\; \varphi_2}{\Gamma; \Delta \vdash t_1 \oplus t_2 \rightsquigarrow t_1' \oplus t_2' : \widehat{\mathbf{bool}} \;\&\; \varphi_1 \cup \varphi_2} \text{ [T-Op]}$$

$$\frac{\Gamma; \Delta \vdash t_1 \rightsquigarrow t_1' : \widehat{\tau}_1 \;\&\; \varphi_1 \quad \Gamma; \Delta \vdash t_2 \rightsquigarrow t_2' : \widehat{\tau}_2 \;\&\; \varphi_2}{\Gamma; \Delta \vdash t_1\ \mathbf{seq}\ t_2 \rightsquigarrow t_1'\ \mathbf{seq}\ t_2' : \widehat{\tau}_2 \;\&\; \varphi_1 \cup \varphi_2} \text{ [T-Seq]}$$

$$\frac{\Gamma; \Delta \vdash t_1 \rightsquigarrow t_1' : \widehat{\mathbf{bool}} \;\&\; \varphi_1 \quad \Gamma; \Delta \vdash t_2 \rightsquigarrow t_2' : \widehat{\tau}_2 \;\&\; \varphi_2 \quad \Gamma; \Delta \vdash t_3 \rightsquigarrow t_3' : \widehat{\tau}_3 \;\&\; \varphi_3}{\Gamma; \Delta \vdash \mathbf{if}\ t_1\ \mathbf{then}\ t_2\ \mathbf{else}\ t_3 \rightsquigarrow \mathbf{if}\ t_1'\ \mathbf{then}\ t_2'\ \mathbf{else}\ t_3' : \widehat{\tau}_2 \sqcup \widehat{\tau}_3 \;\&\; \varphi_1 \cup \varphi_2 \cup \varphi_3} \text{ [T-If]}$$

$$\frac{}{\Gamma; \Delta \vdash []_\tau \rightsquigarrow []_\tau : \bot_\tau \;\&\; \emptyset} \text{ [T-Nil]}$$

$$\frac{\Gamma; \Delta \vdash t_1 \rightsquigarrow t_1' : \widehat{\tau}_1 \;\&\; \varphi_1 \quad \Gamma; \Delta \vdash t_2 \rightsquigarrow t_2' : \left[\widehat{\tau}_1'\langle\varphi_1'\rangle\right] \;\&\; \varphi_2}{\Gamma; \Delta \vdash t_1 :: t_2 \rightsquigarrow t_1' :: t_2' : \left[\widehat{\tau}_1 \sqcup \widehat{\tau}_1'\langle\varphi_1 \cup \varphi_1'\rangle\right] \;\&\; \varphi_2} \text{ [T-Cons]}$$

$$\frac{\begin{array}{c}\Gamma; \Delta \vdash t_1 \rightsquigarrow t_1' : [\tau_1\langle\varphi_1\rangle] \;\&\; \varphi_1' \quad \Gamma; \Delta \vdash t_2 \rightsquigarrow t_2' : \widehat{\tau}_2 \;\&\; \varphi_2 \\ \Gamma, x_1 : \widehat{\tau}_1 \;\&\; \varphi_1, x_2 : [\tau_1\langle\varphi_1\rangle] \;\&\; \varphi_1'; \Delta \vdash t_3 \rightsquigarrow t_3' : \widehat{\tau}_3 \;\&\; \varphi_3\end{array}}{\begin{array}{l}\Gamma; \Delta \vdash \mathbf{case}\ t_1\ \mathbf{of}\ \{[] \mapsto t_2; x_1 :: x_2 \mapsto t_3\} \rightsquigarrow \\ \qquad \mathbf{case}\ t_1'\ \mathbf{of}\ \{[] \mapsto t_2'; x_1 :: x_2 \mapsto t_3'\} : \widehat{\tau}_2 \sqcup \widehat{\tau}_3 \;\&\; \varphi_1' \cup \varphi_2 \cup \varphi_3\end{array}} \text{ [T-Case]}$$

Figure 4: Syntax-directed type elaboration system ($\Gamma; \Delta \vdash t \rightsquigarrow t' : \widehat{\tau} \;\&\; \varphi$)