LinML

Ada Vienot

1 Typing

The types in LinML have the following syntax :

$$
\begin{array}{rcll}
\tau, \sigma & ::= & 0 \mid 1 \mid \top & \text{(type constants)} \\
& \mid & a & \text{(type variables)} \\
& \mid & \tau \multimap \sigma & \text{(linear abstraction)} \\
& \mid & \tau \to \sigma & \text{(unchecked abstraction)} \\
& \mid & \tau + \sigma & \text{(additive disjunction)} \\
& \mid & \tau * \sigma & \text{(multiplicative conjunction)} \\
& \mid & \tau \,\&\, \sigma & \text{(multiplicative disjunction)} \\
& \mid & \tau! & \text{(of course)}
\end{array}
$$

Figure 1: LinML types

1.1 Typing rules

We define the inductive predicate exp over types as follows :

$$
\exp(\tau) = \begin{cases} \text{true} & \text{if } \tau = * \text{ or } \tau = !\sigma \\ \text{false} & \text{elsewise} \end{cases}
$$

In all the following, typing contexts $\Gamma; \Delta$ are to be seen as a pair with :

- Γ a multiset of pairs $(x : \tau)$, with x a variable and τ a type such that $\exp(\tau) = \text{false}$
- Δ a set of pairs $(x : \tau)$, with x a variable and τ a type such that $\exp(\tau) = \text{true}$

Intuitively, Γ contains the linear bindings, and Δ contains the exponential bindings.

Since $\forall \tau, \exp(\tau) = \mathit{true}$ implies the existence of context weakening and contraction rules for τ, we treat the context Δ as the kind of contexts we manipulate in intuitionistic sequent calculus, that is : they can be duplicated, erased, and one binding is allowed to erase another. This is not the case of the context Γ containing linear bindings.

1.1.1 Terms

Let t, u be terms, τ, σ types, and $(\Gamma; \Delta), (\Gamma'; \Delta')$ typing contexts. The typing judgements for terms have the following shape :

$$
\Gamma; \Delta \vdash t : \tau \Rightarrow \Gamma'; \Delta'
$$

$\Gamma'; \Delta'$ is the typing context after consuming the necessary bindings to type $t : \tau$.

The typing rules are the following:

$$\frac{}{\Gamma; \Delta \vdash * : 1 \Rightarrow \Gamma; \Delta}\ (1) \qquad \frac{}{\Gamma; \Delta \vdash \langle\rangle : \top \Rightarrow \Gamma; \Delta}\ (\top)$$

$$\frac{(x:\tau) \in \Gamma \qquad \exp(\tau)}{\Gamma; \Delta \vdash x : \tau \Rightarrow (\Gamma \setminus (x:\tau)); \Delta}\ (\text{vlin}) \qquad \frac{(x:\tau) \in \Delta \qquad \neg \exp(\tau)}{\Gamma; \Delta \vdash x : \tau \Rightarrow \Gamma; \Delta}\ (\text{vexp})$$

$$\frac{\Gamma; \Delta \vdash t : \tau \Rightarrow \Gamma'; \_ \qquad \Gamma'; \Delta' \vdash u : \sigma \Rightarrow \Gamma''; \_}{\Gamma; \Delta \vdash (t, u) : \tau * \sigma \Rightarrow \Gamma''; \Delta}\ (\otimes)$$

$$\frac{\Gamma; \Delta \vdash t : \tau \Rightarrow \Gamma'; \_ \qquad \Gamma; \Delta \vdash u : \sigma \Rightarrow \Gamma''; \_ \quad \Gamma' = \Gamma''}{\Gamma; \Delta \vdash \langle t, u\rangle : \tau \mathbin{\&} \sigma \Rightarrow \Gamma''; \Delta}\ (\&)$$

$$\frac{\Gamma; \Delta \vdash t : \tau \Rightarrow \Gamma'; \_}{\Gamma; \Delta \vdash (t :> \_ + \sigma) : \tau + \sigma \Rightarrow \Gamma'; \Delta}\ (\oplus\text{-l}) \qquad \frac{\Gamma; \Delta \vdash u : \sigma \Rightarrow \Gamma'; \_}{\Gamma; \Delta \vdash (u :> \sigma + \_) : \tau + \sigma \Rightarrow \Gamma'; \Delta}\ (\oplus\text{-r})$$

$$\frac{\Gamma; \Delta \vdash t : \tau \Rightarrow \Gamma'; \Delta \quad \Gamma = \Gamma'}{\Gamma; \Delta \vdash !t : !\tau \Rightarrow \Gamma; \Delta}\ (!) \qquad \frac{\Gamma; \Delta \vdash !t : !\tau \Rightarrow \Gamma'; \Delta}{\Gamma; \Delta \vdash !!t : !\tau \Rightarrow \Gamma'; \Delta}\ (!\text{-dig})$$

$$\frac{\Gamma, x : \tau \vdash t : \sigma \Rightarrow \Gamma}{\Gamma \vdash \mathbf{fun}\ (x : \tau) \multimap t : \tau \multimap \sigma \Rightarrow \Gamma}\ (\multimap) \qquad \frac{\Gamma \vdash t : \sigma \multimap \tau \qquad \Gamma' \vdash t' : \sigma}{\Gamma, \Gamma' \vdash tt' : \tau}\ (\multimap\text{-app})$$

$$\frac{\Gamma, \Delta, \Delta' \vdash t : \sigma \qquad \Gamma, \Delta, x : \sigma \vdash t' : \tau \qquad (x : \sigma) \notin \Gamma}{\Gamma \vdash \mathbf{give}\ x = t\ \mathbf{in}\ t' : \tau}\ (\text{give})$$

$$\frac{\Gamma \vdash \mathbf{give}\ x = !t\ \mathbf{in}\ t' : \tau}{\Gamma \vdash \mathbf{let}\ x = t\ \mathbf{in}\ t' : \tau}\ (\text{let})$$

Figure 2: LinML terms typing rules

1.1.2 Patterns