A Linear Dependent Type Theory

Zhaohui Luo
Royal Holloway
University of London

Yu Zhang
Institute of Software
Chinese Academy of Sciences

Linear types and dependent types

- Linear types (Girard 1987): $A \multimap B$
- Dependent types (Martin-Löf 1970s): $\Pi x{:}A.B[x]$
- How to combine them?
  - In most of existing work (Pfenning et al 2002, Krishnaswami et al 2015, Vákár 2015)
    - $B[x]$ only when x is intuitionistic.
    - Hence it is possible to separate intuitionistic Γ and linear Δ: $\Gamma; \Delta \mid\!- a : A$
    - Δ depends on Γ, but not the other way around.
  - McBride (2016)
    - "Prices" in contextual entries and typing and allow type dependency on 0-priiced variables – discussion later.
    - Independent with this work (we became aware of Conor's work only two weeks ago – detailed comparison due.)
- This paper: LDTT, where types can depend on linear variables.

LDTT: Linear and Intuitionistic Variables

- Contexts are sequences of two forms of entries:

  $x{:}A,\ y{::}B[x],\ z{:}C[x,y],\ \ldots$

  - Intuitionistic variables $x : A$
  - Linear variables $y :: B$
- Types dependent on linear variables
  - Example: $x{::}A,\ f : A \multimap A \vdash Eq_A(f\ x,\ x)$ type

Intuitionistic Π-types

$$\frac{\Gamma,\ x:A \vdash B\ \mathit{type}}{\Gamma \vdash \Pi x{:}A.B\ \mathit{type}} \qquad \frac{\Gamma,\ x:A \vdash b:B}{\Gamma \vdash \lambda x{:}A.b : \Pi x{:}A.B} \qquad \frac{\Gamma \vdash f : \Pi x{:}A.B \quad \overline{\Gamma} \vdash a : A}{\Gamma \vdash f(a) : [a/x]B}$$

- $\overline{\Gamma}$ – the intuitionistic part of context Γ
  - $\overline{\Gamma} = \Gamma \setminus FV_{LD}(\Gamma)$ – removing the linear dependent variables
    - $FV_{LD}(\langle\rangle) = \varnothing$
    - $FV_{LD}(\Gamma,x{:}A) = FV_{LD}(\Gamma)$ if $FV(A) \cap FV_{LD}(\Gamma) = \varnothing$;
      $= FV_{LD}(\Gamma) \cup \{x\}$ otherwise
    - $FV_{LD}(\Gamma,x{::}A) = FV_{LD}(\Gamma) \cup \{x\}$
  - Example: $\Gamma \equiv x{:}A,\ y{::}B,\ z{:}C$
    $\overline{\Gamma} \equiv x{:}A,\ z{:}C$ if $y \notin FV(C)$
    $\overline{\Gamma} \equiv x{:}A,$ if $y \in FV(C)$

Linear Π-types

$$\frac{\Gamma,\ x{::}A \vdash B\ type}{\Gamma \vdash \Pi x{::}A.B\ type} \qquad \frac{\Gamma,\ x{::}A \vdash b : B}{\Gamma \vdash \lambda x{::}A.b : \Pi x{::}A.B} \qquad \frac{\Gamma \vdash f : \Pi x{::}A.B \quad \Delta \vdash a : A \quad Merge(\Gamma;\Delta)\downarrow}{Merge(\Gamma;\Delta) \vdash f\ a : [a/x]B}$$

- Merge(Γ;Δ) is only defined if
  - $\overline{\Gamma} \equiv \overline{\Delta}$ (the intuitionistic parts are the same)
  - $FV_{LD}(\Gamma) \cap FV_{LD}(\Delta) = \emptyset$ (Γ/Δ do not share linear dependent variables)
- When the above are the case, Merge is defined as:

(a) $Merge(\langle\rangle;\langle\rangle) = \langle\rangle$.

(b) If $x \in FV_{LD}(\Gamma) \cup FV_{LD}(\Delta)$,

$$Merge(\Gamma, x\bar{:}A;\ \Delta) = Merge(\Gamma;\ \Delta, x\bar{:}A) = Merge(\Gamma;\Delta), x\bar{:}A,$$

where $\bar{:}$ is either $:$ or $::$.

(c) $Merge(\Gamma, x{:}A;\ \Delta, x{:}A) = Merge(\Gamma;\Delta), x{:}A$.

- Example: $\Gamma \equiv x{:}A,\ y_1{::}B_1,\ z{:}C$

  $\Delta \equiv x{:}A,\ y_2{::}B_2,\ z{:}C$

  $Merge(\Gamma;\Delta) \equiv x{:}A,\ z{:}C,\ y_1{::}B_1,\ y_2{::}B_2$

Note: $y_1 \neq y_2$ and $y_1, y_2 \notin FV(C)$ for otherwise, Merge(Γ;Δ) would be undefined.

Equality Types

- Formation rule

$$\frac{\Gamma \vdash a : A \quad \Delta \vdash b : A \quad merge(\Gamma;\Delta)\downarrow}{merge(\Gamma;\Delta) \vdash Eq_A(a,b)\ type}$$

- merge(Γ;Δ) is defined only when var-sharing is OK:

  x?A∈Γ, x?B∈Δ ➔ A≡B and ? is both : or both ::

- merge(Γ;Δ) is defined as

$$\text{(a)}\quad merge(\Gamma;\langle\rangle) = \Gamma.$$

$$\text{(b)}\quad merge(\Gamma;\ x\bar{:}A,\Delta) = \begin{cases} merge(\Gamma;\Delta) & \text{if } x \in FV(\Gamma) \\ merge(\Gamma, x\bar{:}A;\ \Delta) & \text{otherwise} \end{cases}$$

- Examples:
  - x::A, f : A⊸A ⊢ f x : A and x::A ⊢ x : A ➔ x::A, f : A⊸A ⊢ Eq_A(f x, x) type
  - x::A ⊢ x : A and y::A ⊢ y : A ➔ x::A, y::A ⊢ Eq(x,y) type

❖ Introduction and elimination rules

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash refl(a) : Eq_A(a,a)}$$

$$\frac{\Gamma \vdash p : Eq_A(a,b) \quad \Delta \vdash q : B[a] \quad Merge(\Gamma;\Delta), x\bar{:}A \vdash B[x]\ type\ (\bar{:} \in \{:, ::\}) \quad Merge(\Gamma;\Delta)\downarrow}{Merge(\Gamma;\Delta) \vdash subst(x.B,p,q) : B[b]}$$

Variable Typing

$$\frac{\Gamma,\ x\bar{:}A,\ \Gamma'\ valid \quad (\text{for all } y{::}\Gamma_y \in \Gamma,\ y \in D_\Gamma(x)) \quad \Gamma' \text{ intuitionistic}}{\Gamma,\ x\bar{:}A,\ \Gamma' \vdash x : A} \quad (\bar{:} \in \{:, ::\})$$

where

- *Γ′ intuitionistic* means that it does not have linear ::-entries
- $D_\Gamma(x)$ is defined as:
  - $x \in D_\Gamma(x)$;
  - For any $y \in D_\Gamma(x)$, $FV(\Gamma_y) \subseteq D_\Gamma(x)$.

- Examples:
  - Judgements derivable intuitionistically are derivable.
  - $x{::}A, y{:}B(x) \vdash x{:}A$ and $x{::}A, y{:}B(x) \vdash y{:}B(x)$ are derivable since $x \in B(x)$.
  - $x{::}A, x'{::}A, y{:}B(x) \vdash y : B(x)$ is *not* derivable if $x' \notin B(x)$.

Other Rules (for completeness)

- Context validity

$$(C_1)\ \frac{}{\langle\rangle\ valid} \qquad (C_2)\ \frac{\Gamma \vdash A\ type \quad x \notin FV(\Gamma)}{\Gamma,\ x : A\ valid} \qquad (C_3)\ \frac{\Gamma \vdash A\ type \quad x \notin FV(\Gamma)}{\Gamma,\ x{::}A\ valid}$$

- Context permutation rule:

$$\frac{\Gamma \vdash a : A \quad \Delta\ valid}{\Delta \vdash a : A} \quad (\Delta \in perm(\Gamma))$$

- Conversions and conversion rule:

$$\frac{\Gamma \vdash a : A \quad \Gamma \vdash B\ type}{\Gamma \vdash a : B} \quad (A \simeq B)$$

Intuitionistic Π-*types* (Conversion: $(\lambda x{:}A.b)(a) \simeq [a/x]b$)

Linear Π-*types* (Conversion: $(\lambda x{::}A.b)\ a \simeq [a/x]b$)

Equality types (Conversion: $subst(refl(a), q) \simeq q$)

Weak Linearality

- *Defn (essential occurrences)* Let *Γ |- a:A*. The multiset $E_\Gamma(a)$ of variables essentially occurring in *a* under *Γ* is inductively defined as follows (Eq-types omitted):
  - Variable typing: $E_{\Gamma,\ x\bar{:}A,\ \Gamma'}(x) = D_{\Gamma,\ x\bar{:}A,\ \Gamma'}(x)$
  - λ-typing: $E_\Gamma(\lambda x\bar{:}A.b) = E_{\Gamma,x\bar{:}A}(b)\backslash\{x\}$
  - Intuitionistic applications: $E_\Gamma(f(a)) = E_\Gamma(f) \cup E_{\overline{\Gamma}}(a)$
  - Linear applications: $E_{Merge(\Gamma;\Delta)}(f\ a) = E_\Gamma(f) \cup E_\Delta(a)$
- *Theorem (weak linearality)*

  In LDTT, every linear variable occurs essentially for exactly once in a well-typed term. Formally,

  Γ, y::B, Γ′ |- a : A ➔ $y \in E_{\Gamma,y::B,\Gamma'}(a)$ only once.

Implementation

- Type checking algorithm
  - Follows the traditional algorithm for type inference/checking.
  - Decidability, if assuming meta-theoretic results (expected).
- Prototype implementation in Haskell
  - Merging oprns correspond to splitting oprns.
  - Available online: https://github.com/yveszhang/ldtyping

Related Work

- Work on linearity in dependent types
  - Eg, (Pfenning et al, I&C02), (Krishnaswami et al, POPL15), (Vákár, FoSSaCS 15)
  - Lambek calculus with dependent types (Luo, TYPES 2015)
  - Types in all above are non-dependent on linear/Lambek variables
- McBride 2016 (Walder Festschrift)
  - More general setting: considering "prices" in {0,1,w}:

    $$\rho_1 x_1 : A_1, \ldots \rho_n x_n : A_n \mid\!- \rho\ a{:}A$$

    and different Π-types $(\rho x{:}A) \to B$:
    - $(\omega x{:}A) \to B$ corresponds to intuitionistic Π-types
    - $(1x{:}A) \to B$ corresponds to linear Π-types
  - Type dependency B[x] only on "0-priced" variables x.
  - Independent with the current work and comparison to be done.

Future Work

- LDTT: allowing types to depend on linear variables
  - Simplicity
    - LDTT gives a “straightforward” extension with linearality
    - cf, McBride’s work, analysis to be done
  - Examples of reasoning
    - to be done with our prototype implementation
- Extension to other linear/Lambek type constructors