A Well-typed Interpreter for a Dependently Typed Language

Nils Anders Danielsson

Chalmers

February 21, 2007

Introduction

- Well-typed representation of dependently typed lambda calculus (no raw terms).
- Interpreter (= normalisation proof).
- Implemented in AgdaLight.

Well-typed language

$$
\begin{aligned}
&\textbf{data } \mathit{Ty} : * \textbf{ where}\\
&\quad \mathit{Nat}' : \mathit{Ty}\\
&\quad \mathit{Bool}' : \mathit{Ty}
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{data } \mathit{Op} : \mathit{Ty} \longrightarrow * \textbf{ where}\\
&\quad \mathit{plus} : \mathit{Op}\ \mathit{Nat}'\\
&\quad \mathit{and} : \mathit{Op}\ \mathit{Bool}'
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{data } \mathit{Term} : \mathit{Ty} \longrightarrow * \textbf{ where}\\
&\quad \mathit{nat} : \mathit{Nat} \longrightarrow \mathit{Term}\ \mathit{Nat}'\\
&\quad \mathit{bool} : \mathit{Bool} \longrightarrow \mathit{Term}\ \mathit{Bool}'\\
&\quad \mathit{op} : \{\sigma : \mathit{Ty}\} \longrightarrow \mathit{Op}\ \sigma\\
&\qquad \longrightarrow \mathit{Term}\ \sigma \longrightarrow \mathit{Term}\ \sigma \longrightarrow \mathit{Term}\ \sigma
\end{aligned}
$$

Interpreter which cannot get stuck

$$
\begin{aligned}
&\mathit{Sem} : \mathit{Ty} \;\rightarrow\; * \\
&\mathit{Sem}\ \mathit{Nat}' \;= \mathit{Nat} \\
&\mathit{Sem}\ \mathit{Bool}' = \mathit{Bool}
\end{aligned}
$$

$$
\begin{aligned}
&[\![\cdot]\!] : \mathit{Term}\ \sigma \;\rightarrow\; \mathit{Sem}\ \sigma \\
&[\![\mathit{nat}\ n]\!] \qquad = n \\
&[\![\mathit{bool}\ b]\!] \qquad = b \\
&[\![\mathit{op}\ o\ t_1\ t_2]\!] = \mathit{fun}\ o\ [\![t_1]\!]\ [\![t_2]\!] \\
&\quad \textbf{where} \\
&\quad \mathit{fun} : \mathit{Op}\ \sigma \;\rightarrow\; (\mathit{Sem}\ \sigma \;\rightarrow\; \mathit{Sem}\ \sigma \;\rightarrow\; \mathit{Sem}\ \sigma) \\
&\quad \mathit{fun}\ \mathit{plus} = (+) \\
&\quad \mathit{fun}\ \mathit{and}\ = (\wedge)
\end{aligned}
$$

Object language

- Variant of Martin-Löf's logical framework.
- Variables (de Bruijn indices).
- Explicit substitutions.
- All definitions are mutually recursive.

Contexts

$$\begin{array}{l}\textbf{data}\ \mathit{Ctxt} : * \ \textbf{where}\\ \quad \varepsilon \quad : \mathit{Ctxt}\\ \quad (\triangleright) : (\Gamma : \mathit{Ctxt}) \rightarrow \mathit{Ty}\ \Gamma \rightarrow \mathit{Ctxt}\end{array}$$

$$\frac{}{\varepsilon\ \mathit{context}} \qquad \frac{\Gamma\ \mathit{context} \qquad \Gamma \vdash \tau\ \mathit{type}}{\Gamma \triangleright \tau\ \mathit{context}}$$

Types

$$\begin{aligned}
&\textbf{data } \mathit{Ty} : \mathit{Ctxt} \to * \textbf{ where}\\
&\quad \star : \mathit{Ty}\ \Gamma\\
&\quad \mathit{El} : \Gamma \vdash \star \to \mathit{Ty}\ \Gamma\\
&\quad \Pi : (\tau : \mathit{Ty}\ \Gamma) \to \mathit{Ty}\ (\Gamma \triangleright \tau) \to \mathit{Ty}\ \Gamma
\end{aligned}$$

$$\frac{}{\Gamma \vdash \star\ \mathit{type}} \qquad \frac{\Gamma \vdash t : \star}{\Gamma \vdash \mathit{El}\ t\ \mathit{type}}$$

$$\frac{\Gamma \vdash \tau_1\ \mathit{type} \qquad \Gamma \triangleright \tau_1 \vdash \tau_2\ \mathit{type}}{\Gamma \vdash \Pi\ \tau_1\ \tau_2\ \mathit{type}}$$

Types

$$\textbf{data}\ \mathit{Ty} : \mathit{Ctxt} \to * \ \textbf{where}$$
$$\star : \mathit{Ty}\ \Gamma$$
$$\mathit{El} : \Gamma \vdash \star \to \mathit{Ty}\ \Gamma$$
$$\Pi : (\tau : \mathit{Ty}\ \Gamma) \to \mathit{Ty}\ (\Gamma \triangleright \tau) \to \mathit{Ty}\ \Gamma$$

$$(/) : \mathit{Ty}\ \Gamma \to \Gamma \Rightarrow \Delta \to \mathit{Ty}\ \Delta$$
$$\star \ / \ \rho = \star$$
$$\mathit{El}\ t \ / \ \rho = \mathit{El}\ (t \ /_{\vdash}\ \rho)$$
$$\Pi\ \tau_1\ \tau_2 \ / \ \rho = \Pi\ (\tau_1 \ / \ \rho)\ (\tau_2 \ / \ \rho \uparrow \tau_1)$$

Terms

$$\frac{\Gamma \vdash v : \tau}{\Gamma \vdash var\ v : \tau} \qquad \frac{\Gamma \triangleright \tau_1 \vdash t : \tau_2}{\Gamma \vdash \lambda\ t : \Pi\ \tau_1\ \tau_2}$$

$$\frac{\Gamma \vdash t_1 : \Pi\ \tau_1\ \tau_2 \qquad \Gamma \vdash t_2 : \tau_1}{\Gamma \vdash t_1 @ t_2 : \tau_2\ /\ sub\ t_2}$$

$$\frac{\Gamma \vdash t : \tau_1 \qquad eq : \tau_1 =_\star \tau_2}{\Gamma \vdash t ::_\vdash eq : \tau_2}$$

$$\frac{\Gamma \vdash t : \tau \qquad \rho : \Gamma \Rightarrow \Delta}{\Delta \vdash t \,/_\vdash\, \rho : \tau\ /\ \rho}$$

Variables

$(\ni)$: $(\Gamma : Ctxt) \rightarrow Ty\ \Gamma \rightarrow *$

vz : $\Gamma \triangleright \sigma \ni \sigma \,/\, wk\ \sigma$

Variable zero.

σ is in Γ, not $\Gamma \triangleright \sigma$, so it needs to be weakened.

$vs\ v$ The variable after v.

$(::_{\ni})$ Conversion rule.

Substitutions

$(\Rightarrow) \; : \; Ctxt \;\rightarrow\; Ctxt \;\rightarrow\; *$

sub Single term substitutions ($[vz := t]$).

wk Weakenings.

$(\uparrow)$ Lifting.

id Identity.

$(\odot)$ Composition.

Equality

- Congruence.
- β- and η-rules.
- Evaluation rules for $(/_{\vdash})$.
- Casts $(::)$ can be removed.

Interpreter

A choice

- Use general recursion.
  - Relatively easy.
  - Partially correct.
  - No immediate guarantees about termination.
- Use structural recursion.
  - Harder.
  - Totally correct.

Here: structural recursion using normalisation by evaluation.

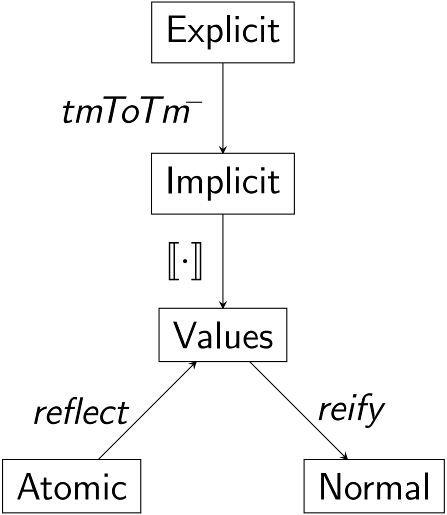
Interpreter

A choice

- Use general recursion.
  - Relatively easy.
  - Partially correct.
  - No immediate guarantees about termination.
- Use structural recursion.
  - Harder.
  - Totally correct.

Here: structural recursion using normalisation by evaluation.

Overview



Explicit
tmToTm⁻
Implicit
⟦·⟧
Values
reflect
reify
Atomic
Normal

Types ensure soundness

- $\mathit{Tm}^{-}, \mathit{Val}, \mathit{Atom}, \mathit{NF} : \Gamma \vdash \tau \;\rightarrow\; *$
- $\mathit{nf} : \mathit{NF}\ t$ means that nf is a normal form $\beta\eta$-equivalent to t.

$$
\begin{array}{lll}
\mathit{tmToTm}^{-} & : (t : \Gamma \vdash \tau) & \rightarrow \mathit{Tm}^{-}\ t \\
[\![\cdot]\!] & : \mathit{Tm}^{-}\ t \rightarrow \mathit{Env}\ \rho & \rightarrow \mathit{Val}\ (t \mathbin{/_{\vdash}} \rho) \\
\mathit{reflect} & : \mathit{Atom}\ t & \rightarrow \mathit{Val}\ t \\
\mathit{reify} & : \mathit{Val}\ t & \rightarrow \mathit{NF}\ t
\end{array}
$$

Interesting cases

- Π types are represented using functions that perform application.

$$
\begin{array}{l}
\textbf{data}\ \mathit{Val} : \Gamma \vdash \tau\ \rightarrow\ *\ \textbf{where} \\
\quad \Pi_{\mathit{Val}}\ :\ \{ t_1 : \Gamma \vdash \Pi\ \tau_1\ \tau_2 \} \\
\qquad\qquad \rightarrow (f\ :\ (\Gamma' : \mathit{Ctxt}^{+}\ \Gamma) \\
\qquad\qquad\qquad \rightarrow \{ t_2 : \Gamma\ +\!\!+\ \Gamma' \vdash \tau_1\ /\ \mathit{wk}^{*}\ \Gamma' \} \\
\qquad\qquad\qquad \rightarrow (v_2 : \mathit{Val}\ t_2) \\
\qquad\qquad\qquad \rightarrow \mathit{Val}\ ((t_1\ /_{\vdash}\ \mathit{wk}^{*}\ \Gamma')@t_2)) \\
\qquad\qquad \rightarrow \mathit{Val}\ t_1
\end{array}
$$

$$
\begin{array}{l}
[\![\cdot]\!] : \mathit{Tm}^{-}\ t\ \rightarrow\ \mathit{Env}\ \rho\ \rightarrow\ \mathit{Val}\ (t\ /_{\vdash}\ \rho) \\
[\![\lambda^{-}\ t_1^{-}]\!]\gamma = \Pi_{\mathit{Val}}\,(\backslash\Delta'\ v_2\ \rightarrow \\
\quad [\![t_1^{-}]\!](\mathit{wk}^{\star}_{\mathit{Env}}\ \gamma\ \Delta'\ \blacktriangleright_{\mathit{Env}}\ (v_2\ ::_{\mathit{Val}}\ \ldots))\ ::_{\mathit{Val}}\ \ldots\beta\ldots)
\end{array}
$$

Interesting cases

- Π types are represented using functions that perform application.

$$
\begin{aligned}
&\textbf{data } \mathit{Val} : \Gamma \vdash \tau \;\rightarrow\; * \textbf{ where} \\
&\quad \Pi_{\mathit{Val}} \;:\; \{\, t_1 : \Gamma \vdash \Pi\ \tau_1\ \tau_2 \,\} \\
&\qquad \rightarrow (f \;:\; (\Gamma' : \mathit{Ctxt}^{+}\ \Gamma) \\
&\qquad\qquad \rightarrow \{\, t_2 : \Gamma \;+\!\!+\; \Gamma' \vdash \tau_1 \;/\; \mathit{wk}^{*}\ \Gamma' \,\} \\
&\qquad\qquad \rightarrow (v_2 : \mathit{Val}\ t_2) \\
&\qquad\qquad \rightarrow \mathit{Val}\ ((t_1 \;/\!\!\vdash\; \mathit{wk}^{*}\ \Gamma')@t_2)) \\
&\qquad \rightarrow \mathit{Val}\ t_1
\end{aligned}
$$

$$
\begin{aligned}
&[\![\cdot]\!] : \mathit{Tm}^{-}\ t \;\rightarrow\; \mathit{Env}\ \rho \;\rightarrow\; \mathit{Val}\ (t \;/\!\!\vdash\; \rho) \\
&[\![\lambda^{-}\ t_1^{-}]\!]\gamma = \Pi_{\mathit{Val}}\,(\backslash\Delta'\ v_2 \;\rightarrow \\
&\quad [\![t_1^{-}]\!]\,(\mathit{wk}^{\star}_{\mathit{Env}}\ \gamma\ \Delta' \blacktriangleright_{\mathit{Env}} (v_2 \;::_{\mathit{Val}}\; \ldots))\ ::_{\mathit{Val}}\ \ldots\beta\ldots)
\end{aligned}
$$

Interesting cases

$$
\begin{aligned}
&\mathit{reify} : (\tau : \mathit{Ty}\ \Gamma) \to \{t : \Gamma \vdash \tau\} \to \mathit{Val}\ t \to \mathit{NF}\ t\\
&\mathit{reify}\ (\Pi\ \tau_1\ \tau_2)\ (\Pi_{\mathit{Val}}\ f) =\\
&\quad \lambda_{\mathit{NF}}\ (\mathit{reify}\ (\tau_2\ /\ \_\ /\ \_)\\
&\qquad\qquad\quad (f\ (\varepsilon^{+} \rhd^{+} \tau_1)\\
&\qquad\qquad\qquad (\mathit{reflect}\ (\tau_1\ /\ \_)\ (\mathit{var}_{\mathit{At}}\ \mathit{vz})\ ::_{\mathit{Val}}\ \ldots)))\\
&\quad ::_{\mathit{NF}} \ldots \eta \ldots
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{reflect} : (\tau : \mathit{Ty}\ \Gamma) \to \{t : \Gamma \vdash \tau\} \to \mathit{Atom}\ t \to \mathit{Val}\ t\\
&\mathit{reflect}\ (\Pi\ \tau_1\ \tau_2)\ \mathit{at}_1 = \Pi_{\mathit{Val}}\ (\backslash\Gamma'\ v_2 \to\\
&\quad \mathit{reflect}\ (\tau_2\ /\ \_\ /\ \_)\ (\mathit{wk}^{*}_{\mathit{At}}\ \mathit{at}_1\ \Gamma'\ @_{\mathit{At}}\ \mathit{reify}\ (\tau_1\ /\ \_)\ v_2))
\end{aligned}
$$

Type checker

- What about parsing?
- Using the normaliser it is easy to define a type checker.

Discussion

- Looks horribly complicated?
  Remember that the types used are very precise:
  - Hence guiding the programmer.
  - And limiting the amount of possible errors.
- But it's not something you hack up in an afternoon.

Finally. . .

- For more details, see paper: *A Formalisation of a Dependently Typed Language as an Inductive-Recursive Family*.
- Source code available to play with.

Appendix

Terms

$$
\begin{array}{llcll}
\textbf{data}\ (\vdash) : (\Gamma : \mathit{Ctxt}) \rightarrow \mathit{Ty}\ \Gamma \rightarrow * \ \textbf{where} \\
\quad \mathit{var} & : \Gamma \ni \tau & & & \rightarrow \Gamma \vdash \tau \\
\quad \lambda & : \Gamma \triangleright \tau_1 \vdash \tau_2 & & & \rightarrow \Gamma \vdash \Pi\ \tau_1\ \tau_2 \\
\quad (@) & : \Gamma \vdash \Pi\ \tau_1\ \tau_2 & \rightarrow & (t_2 : \Gamma \vdash \tau_1) & \rightarrow \Gamma \vdash \tau_2 \ / \ \mathit{sub}\ t_2 \\
\quad (::_\vdash) & : \Gamma \vdash \tau_1 & \rightarrow & \tau_1 =_\star \tau_2 & \rightarrow \Gamma \vdash \tau_2 \\
\quad (/_\vdash) & : \Gamma \vdash \tau & \rightarrow & (\rho : \Gamma \Rightarrow \Delta) & \rightarrow \Delta \vdash \tau \ / \ \rho
\end{array}
$$

Variables

$$
\begin{array}{lll}
\textbf{data}\ (\ni) : (\Gamma : \mathit{Ctxt}) \to \mathit{Ty}\ \Gamma \to * \ \textbf{where} & & \\
\quad \mathit{vz} & : & \Gamma \rhd \sigma \ni \sigma \,/\, \mathit{wk}\ \sigma \\
\quad \mathit{vs} & : & \Gamma \ni \tau \to \Gamma \rhd \sigma \ni \tau \,/\, \mathit{wk}\ \sigma \\
\quad (::_{\ni}) & : & \Gamma \ni \tau_1 \to \tau_1 =_{\star} \tau_2 \to \Gamma \ni \tau_2
\end{array}
$$

Substitutions

$$
\begin{array}{l}
\textbf{data}\ (\Rightarrow) : \mathit{Ctxt} \ \rightarrow \ \mathit{Ctxt} \ \rightarrow \ * \ \textbf{where} \\
\quad \mathit{sub} : \Gamma \vdash \tau \qquad \rightarrow \Gamma \triangleright \tau \Rightarrow \Gamma \\
\quad \mathit{wk} \ : (\sigma : \mathit{Ty}\ \Gamma) \rightarrow \Gamma \Rightarrow \Gamma \triangleright \sigma \\
\quad \mathit{id} \ \ : \Gamma \Rightarrow \Gamma \\
\quad (\odot) : \Gamma \Rightarrow \Delta \ \rightarrow \ \Delta \Rightarrow X \ \rightarrow \ \Gamma \Rightarrow X \\
\quad (\uparrow) \ : \quad (\rho : \Gamma \Rightarrow \Delta) \ \rightarrow \ (\sigma : \mathit{Ty}\ \Gamma) \\
\qquad \rightarrow \ \Gamma \triangleright \sigma \ \Rightarrow \ \Delta \triangleright (\sigma \ / \ \rho)
\end{array}
$$

Term equality

$$
\begin{array}{l}
\textbf{data}\ (=_\vdash) : \Gamma_1 \vdash \tau_1 \ \longrightarrow\ \Gamma_2 \vdash \tau_2 \ \longrightarrow\ * \ \textbf{where} \\
\quad\quad \text{-- Equivalence.} \\
\quad \mathit{refl}_\vdash \quad : (t : \Gamma \vdash \tau) \ \longrightarrow\ t =_\vdash t \\
\quad \mathit{sym}_\vdash \quad : t_1 =_\vdash t_2 \ \longrightarrow\ t_2 =_\vdash t_1 \\
\quad \mathit{trans}_\vdash : t_1 =_\vdash t_2 \ \longrightarrow\ t_2 =_\vdash t_3 \ \longrightarrow\ t_1 =_\vdash t_3 \\
\quad\quad \text{-- Congruence.} \\
\quad \mathit{var}_{\mathit{Cong}} \ : v_1 =_\ni v_2 \ \longrightarrow\ \mathit{var}\ v_1 =_\vdash \mathit{var}\ v_2 \\
\quad \lambda_{\mathit{Cong}} \quad : t_1 =_\vdash t_2 \ \longrightarrow\ \lambda\ t_1 =_\vdash \lambda\ t_2 \\
\quad (@_{\mathit{Cong}}) : t_1^1 =_\vdash t_1^2 \ \longrightarrow\ t_2^1 =_\vdash t_2^2 \ \longrightarrow\ t_1^1 @ t_2^1 =_\vdash t_1^2 @ t_2^2 \\
\quad (/_{\vdash\mathit{Cong}}) : t_1 =_\vdash t_2 \ \longrightarrow\ \rho_1 =_\Rightarrow \rho_2 \ \longrightarrow\ t_1 /_\vdash \rho_1 =_\vdash t_2 /_\vdash \rho_2 \\
\quad \ldots
\end{array}
$$

Term equality

$$
\begin{array}{l}
\textbf{data}\ (=_{\vdash}) : \Gamma_1 \vdash \tau_1\ \rightarrow\ \Gamma_2 \vdash \tau_2\ \rightarrow\ *\ \textbf{where} \\
\quad \text{-- Cast, beta and eta equality.} \\
\quad \begin{array}{lll}
\mathit{castEq}_{\vdash} & : & (t ::_{\vdash} \mathit{eq}) =_{\vdash} t \\
\beta & : & (\lambda\ t_1) @ t_2 =_{\vdash} t_1 \mathbin{/_{\vdash}} \mathit{sub}\ t_2 \\
\eta & : & \quad (t : \Gamma \vdash \Pi\ \tau_1\ \tau_2) \\
 & & \rightarrow\ \lambda\ ((t \mathbin{/_{\vdash}} \mathit{wk}\ \tau_1) @ \mathit{var}\ \mathit{vz}) =_{\vdash} t
\end{array} \\
\quad \ldots
\end{array}
$$

Term equality

$$
\begin{array}{llll}
\multicolumn{4}{l}{\textbf{data}\ (=_\vdash) : \Gamma_1 \vdash \tau_1 \ \rightarrow\ \Gamma_2 \vdash \tau_2 \ \rightarrow\ * \ \textbf{where}} \\
\multicolumn{4}{l}{\quad \text{-- Substitution application axioms.}} \\
\lambda\ t & /_\vdash\ \rho & =_\vdash & \lambda\ (t\ /_\vdash\ \rho \uparrow \tau_1) \\
(t_1 @ t_2) & /_\vdash\ \rho & =_\vdash & (t_1\ /_\vdash\ \rho) @ (t_2\ /_\vdash\ \rho) \\
t & /_\vdash\ id & =_\vdash & t \\
t & /_\vdash\ (\rho_1 \odot \rho_2) & =_\vdash & t\ /_\vdash\ \rho_1\ /_\vdash\ \rho_2 \\
var\ v & /_\vdash\ wk\ \sigma & =_\vdash & var\ (vs\ v) \\
var\ vz & /_\vdash\ sub\ t & =_\vdash & t \\
var\ (vs\ v) & /_\vdash\ sub\ t & =_\vdash & var\ v \\
var\ vz & /_\vdash\ (\rho \uparrow \sigma) & =_\vdash & var\ vz \\
var\ (vs\ v) & /_\vdash\ (\rho \uparrow \sigma) & =_\vdash & var\ v\ /_\vdash\ \rho\ /_\vdash\ wk\ (\sigma\ /\ \rho)
\end{array}
$$

Implicit substitutions

$$
\begin{array}{l}
\textbf{data } Tm^{-} : \Gamma \vdash \tau \;\rightarrow\; * \textbf{ where} \\
\quad var^{-} \;\; : (v : \Gamma \ni \tau) \;\rightarrow\; Tm^{-}\ (var\ v) \\
\quad \lambda^{-} \;\;\; : \quad \{t : \Gamma \triangleright \tau_1 \vdash \tau_2\} \\
\qquad\qquad \rightarrow\; Ty^{-}\ \tau_1 \;\rightarrow\; Tm^{-}\ t \\
\qquad\qquad \rightarrow\; Tm^{-}\ (\lambda\ t) \\
\quad (@^{-}) \;\; : Tm^{-}\ t_1 \;\rightarrow\; Tm^{-}\ t_2 \;\rightarrow\; Tm^{-}\ (t_1 @ t_2) \\
\quad (::_{\vdash^{-}}) : Tm^{-}\ t_1 \;\rightarrow\; t_1 =_{\vdash} t_2 \;\rightarrow\; Tm^{-}\ t_2
\end{array}
$$

$$
\begin{array}{ll}
tm^{-}ToTm & : \{t : \Gamma \vdash \tau\} \;\rightarrow\; Tm^{-}\ t \;\rightarrow\; \Gamma \vdash \tau \\
tm^{-}ToTmEq & : (t^{-} : Tm^{-}\ t) \;\rightarrow\; tm^{-}ToTm\ t^{-} =_{\vdash} t \\
tmToTm^{-} & : (t : \Gamma \vdash \tau) \;\rightarrow\; Tm^{-}\ t
\end{array}
$$

Normal forms

$$\begin{array}{l}
\textbf{data}\ \mathit{Atom} : \Gamma \vdash \tau\ \longrightarrow\ *\ \textbf{where} \\
\quad \begin{array}{lll}
\mathit{var}_{\mathit{At}} & : & (v : \Gamma \ni \tau)\ \longrightarrow\ \mathit{Atom}\ (\mathit{var}\ v) \\
(@_{\mathit{At}}) & : & \mathit{Atom}\ t_1\ \longrightarrow\ \mathit{NF}\ t_2\ \longrightarrow\ \mathit{Atom}\ (t_1 @ t_2) \\
(::_{\mathit{At}}) & : & \mathit{Atom}\ t_1\ \longrightarrow\ t_1 =_{\vdash} t_2\ \longrightarrow\ \mathit{Atom}\ t_2
\end{array}
\end{array}$$

$$\begin{array}{l}
\textbf{data}\ \mathit{NF} : \Gamma \vdash \tau\ \longrightarrow\ *\ \textbf{where} \\
\quad \begin{array}{lll}
\mathit{atom}^{\star}_{\mathit{NF}} & : & \{t : \Gamma \vdash \star\}\ \longrightarrow\ \mathit{Atom}\ t\ \longrightarrow\ \mathit{NF}\ t \\
\mathit{atom}^{\mathit{El}}_{\mathit{NF}} & : & \{t : \Gamma \vdash \mathit{El}\ t'\}\ \longrightarrow\ \mathit{Atom}\ t\ \longrightarrow\ \mathit{NF}\ t \\
\lambda_{\mathit{NF}} & : & \mathit{NF}\ t\ \longrightarrow\ \mathit{NF}\ (\lambda\ t) \\
(::_{\mathit{NF}}) & : & \mathit{NF}\ t_1\ \longrightarrow\ t_1 =_{\vdash} t_2\ \longrightarrow\ \mathit{NF}\ t_2
\end{array}
\end{array}$$

Context extensions

$$\begin{array}{l}
\textbf{data}\ \mathit{Ctxt}^{+}\ (\Gamma : \mathit{Ctxt}) : * \ \textbf{where} \\
\quad \varepsilon^{+} \quad : \mathit{Ctxt}^{+}\ \Gamma \\
\quad (\triangleright^{+}) : (\Gamma' : \mathit{Ctxt}^{+}\ \Gamma) \rightarrow \mathit{Ty}\ (\Gamma +\!\!+\ \Gamma') \rightarrow \mathit{Ctxt}^{+}\ \Gamma
\end{array}$$

$$\begin{array}{l}
(+\!\!+) : (\Gamma : \mathit{Ctxt}) \rightarrow \mathit{Ctxt}^{+}\ \Gamma \rightarrow \mathit{Ctxt} \\
\Gamma +\!\!+\ \varepsilon^{+} \qquad\qquad = \Gamma \\
\Gamma +\!\!+\ (\Gamma' \triangleright^{+} \tau) = (\Gamma +\!\!+\ \Gamma') \triangleright \tau
\end{array}$$

Values

$$
\begin{array}{l}
\textbf{data } \mathit{Val} : \Gamma \vdash \tau \ \to \ * \textbf{ where} \\
\quad (::_{\mathit{Val}}) : \mathit{Val}\ t_1 \ \to\ t_1 =_{\vdash} t_2 \ \to\ \mathit{Val}\ t_2 \\
\quad \star_{\mathit{Val}} \ : \{\, t : \Gamma \vdash \star \,\} \quad \to\ \mathit{Atom}\ t \ \to\ \mathit{Val}\ t \\
\quad \mathit{El}_{\mathit{Val}} \ : \{\, t : \Gamma \vdash \mathit{El}\ t' \,\} \ \to\ \mathit{Atom}\ t \ \to\ \mathit{Val}\ t \\
\quad \Pi_{\mathit{Val}} \ : \quad \{\, t_1 : \Gamma \vdash \Pi\ \tau_1\ \tau_2 \,\} \\
\qquad\qquad \to\ (f \ :\ (\Gamma' : \mathit{Ctxt}^{+}\ \Gamma) \\
\qquad\qquad\qquad\qquad \to\ \{\, t_2 : \Gamma + \!\!\! + \Gamma' \vdash \tau_1 \ / \ \mathit{wk}^{*}\ \Gamma' \,\} \\
\qquad\qquad\qquad\qquad \to\ (v_2 : \mathit{Val}\ t_2) \\
\qquad\qquad\qquad\qquad \to\ \mathit{Val}\ ((t_1 \ /_{\vdash}\ \mathit{wk}^{*}\ \Gamma')@t_2)) \\
\qquad\qquad \to\ \mathit{Val}\ t_1
\end{array}
$$

$$
\begin{array}{l}
(@_{\mathit{Val}}) : \mathit{Val}\ t_1 \ \to\ \mathit{Val}\ t_2 \ \to\ \mathit{Val}\ (t_1 @ t_2) \\
\mathit{wk}^{\star}_{\mathit{Val}} \ : \mathit{Val}\ t \ \to\ (\Gamma' : \mathit{Ctxt}^{+}\ \Gamma) \ \to\ \mathit{Val}\ (t \ /_{\vdash}\ \mathit{wk}^{*}\ \Gamma')
\end{array}
$$

Environments

$$\emptyset \quad : \varepsilon \Rightarrow \Delta$$
$$(\blacktriangleright) \quad : (\rho : \Gamma \Rightarrow \Delta) \to \Delta \vdash \tau \mathrel{/} \rho \to \Gamma \triangleright \tau \Rightarrow \Delta$$

$$\begin{array}{l}
\textbf{data } \mathit{Env} : \Gamma \Rightarrow \Delta \to * \textbf{ where} \\
\quad \emptyset_{\mathit{Env}} \quad : \mathit{Env}\ \emptyset \\
\quad (\blacktriangleright_{\mathit{Env}}) : \quad \{\rho : \Gamma \Rightarrow \Delta\} \to \{t : \Delta \vdash \sigma \mathrel{/} \rho\} \\
\qquad\qquad \to \mathit{Env}\ \rho \to \mathit{Val}\ t \to \mathit{Env}\ (\rho \blacktriangleright t) \\
\quad (::_{\mathit{Env}}) : \mathit{Env}\ \rho_1 \to \rho_1 =_{\Rightarrow} \rho_2 \to \mathit{Env}\ \rho_2
\end{array}$$

$$\mathit{lookup} : (v : \Gamma \ni \tau) \to \mathit{Env}\ \rho \to \mathit{Val}\ (\mathit{var}\ v \mathrel{/_{\vdash}} \rho)$$

Evaluation

$$
\begin{aligned}
&[\![\cdot]\!] : Tm^{-}\ t \rightarrow Env\ \rho \rightarrow Val\ (t \not\vdash \rho)\\
&[\![var^{-}\ v]\!]\,\gamma \quad = lookup\ v\ \gamma\\
&[\![t_1^{-}\ @^{-}\ t_2^{-}]\!]\,\gamma \ = ([\![t_1^{-}]\!]\,\gamma\ @_{Val}\ [\![t_2^{-}]\!]\,\gamma)\ ::_{Val}\ \ldots\\
&[\![t^{-}\ ::_{\vdash^{-}}\ eq]\!]\,\gamma = [\![t^{-}]\!]\,\gamma\ ::_{Val}\ \ldots\\
&[\![\lambda^{-}\ t_1^{-}]\!]\,\gamma \quad = \Pi_{Val}\,(\backslash\Delta'\ v_2 \rightarrow\\
&\quad [\![t_1^{-}]\!]\,(wk^{\star}_{Env}\ \gamma\ \Delta'\ \blacktriangleright_{Env}\ (v_2\ ::_{Val} \ldots))\ ::_{Val}\ \ldots\beta\ldots)
\end{aligned}
$$

Normalisation

$$id_{Env} : (\Gamma : Ctxt) \rightarrow Env\ (id\ \Gamma)$$

$$normalise : (t : \Gamma \vdash \tau) \rightarrow NF\ t$$
$$normalise\ t = reify\ \_\ ([\![tmToTm^{-}\ t]\!]\ id_{Env}\ ::_{Val} \ldots)$$

$$normaliseEq : (t : \Gamma \vdash \tau) \rightarrow nfToTm\ (normalise\ t) =_{\vdash} t$$
$$normaliseEq\ t = nfToTmEq\ (normalise\ t)$$