an Intensional Dependent Type Theory with Type-in-Type and Recursion

February 23, 2021

1 Language

1.1 Surface Language

l			position identifier
Γ	::=	$\lozenge \mid \Gamma, x : M$	var contexts
m, n, h, M, N, H, P	::=	x	expressions: variable
	|	$m ::_l M$	type annotation
	|	$\star$	type universe
	|	$\Pi x : M_l . N_{l'}$	function type
	|	$\mathsf{fun}\, f.\, x.\, m \mid m\, {}_l n$	function constructor, eliminator
v	::=	x	values
	|	$\star \mid \Pi x : M.N$	type values
	|	$\mathsf{fun}\, f.\, x.\, n$	function values

1.2 Cast Language

H	::=	$\lozenge \mid H, x : A$	var contexts
a_h, b_h, c_h	::=	x	
	|	$\star$	
	|	$\Pi x : A.B$	
	|	$\mathsf{fun}\, f.\, x.a \mid b\, a$	
e	::=	$A \mid e =_{l,o} A$	type equality chain
a, b, c, A, B, C	::=	$\star \mid \Pi x : A.B$	
	|	$a_h :: e$	
o	::=	$. \mid o.arg$	observation
	|	$o.bod[a]$	

2 Definitions

2.1 Substitution

$$
\begin{array}{rll}
\star\,[x \coloneqq a] & = \star & b\,[x \coloneqq a] \to c \\
(\Pi x : A.B)\,[x \coloneqq a] & = \Pi x : A\,[x \coloneqq a]\,.B\,[x \coloneqq a] & \\
(x :: A =_{l,o} e)\,[x \coloneqq a_h :: e'] & = a_h :: e' =_{l,o} e\,[x \coloneqq a_h :: e'] & \\
(y :: e)\,[x \coloneqq a] & = y :: e\,[x \coloneqq a_h :: e'] & \\
(b_h :: e)\,[x \coloneqq a] & = b_h\,[x \coloneqq a] :: e\,[x \coloneqq a] & \\
\star\,[x \coloneqq a] & = \star & b_h\,[x \coloneqq a] \dashrightarrow c_h \\
(\Pi x : A.B)\,[x \coloneqq a] & = \Pi x : A\,[x \coloneqq a]\,.B\,[x \coloneqq a] & \\
(\mathsf{fun}\, f.\, y.b)\,[x \coloneqq a] & = \mathsf{fun}\, f.\, y.b\,[x \coloneqq a] & \\
(b\,c)\,[x \coloneqq a] & = b\,[x \coloneqq a]\;c\,[x \coloneqq a] & \\
(e =_{l,o} A)\,[x \coloneqq a] & e\,[x \coloneqq a] =_{l,o[x \coloneqq a]} A\,[x \coloneqq a] & e\,[x \coloneqq a] \to e' \\
B\,[x \coloneqq a] & B\,[x \coloneqq a] & \\
.\,[x \coloneqq a] & . & o\,[x \coloneqq a] \to o' \\
(o.arg)\,[x \coloneqq a] & o\,[x \coloneqq a]\,.arg & \\
(o.bod[b])\,[x \coloneqq a] & o\,[x \coloneqq a]\,.bod[b\,[x \coloneqq a]] &
\end{array}
$$

and for contexts.

2.2 lookup

$$
\begin{array}{rll}
A\uparrow & = A & \text{apparent type} \\
e =_{l,o} A\uparrow & = A & \\
A\downarrow & = A & \text{constructor type} \\
e =_{l,o} A\downarrow & = e\downarrow &
\end{array}
$$

2.3 Casts

occasionally we will use the shorthand $A :: e$ to inject additional casts into A,

$$
\begin{array}{rl}
(A :: B) :: (B' =_{l,o} e) & = A :: B =_{l,o} e \\
(A :: e' =_{l,o} B) :: (B' =_{l,o} e) & = A :: e' =_{l,o} e
\end{array}
$$

3 Judgments

$H \vdash n\,Elab\,a$	Infer cast
$H \vdash n\,Elab_{A,l}\,a$	Check cast*
$H \vdash$	well formed context (not presented)
$H \vdash a : A$	apparent type
$H \vdash e : \overline{\star}$	well formed casts
$H \vdash a \equiv a' : A$	
$H \vdash a \Rrightarrow_{*} a' : A$	
$H \vdash a \Rrightarrow a' : A$	
$H \vdash e \Rrightarrow e' : \overline{\star}$	par reductions
$H \vdash o \Rrightarrow o'$	
$A \sim A'$	same except for observations and evidence
$e \sim e'$	
$e\,Elim_{\star}$	concrete elimination
$e\,Elim_{\Pi} x : e_A.e_b$	

3.1 Head Judgments

It is helpful to present some judgments that only consider head form, this avoids some bookkeeping with casts

$H \vdash a_h : A$	head type
$H \vdash a_h \Rrightarrow a : A$	
$H \vdash a_h \sim a'_h$	

3.2 Elaboration

3.2.1 Infer

$$\frac{x : A \in H}{H \vdash x\,Elab\,x :: A}$$

$$\frac{H \vdash M\,Elab_{\star,l}\,C \quad H \vdash m\,Elab_{C,l}\,a}{H \vdash m ::_l M\,Elab\,a}$$

$$\frac{H \vdash}{H \vdash \star\,Elab\,\star}$$

$$\frac{H \vdash M\,Elab_{\star,l}\,A \quad H, x : A \vdash N\,Elab_{\star,l'}\,B}{H \vdash \Pi x : M_l.N_{l'}\,Elab\,\Pi x : A.B}$$

$$\frac{H \vdash m\,Elab\,b_h :: e \quad \Pi x : A.B = e \uparrow \quad H \vdash n\,Elab_{A,l}\,a}{H \vdash m\,{}_l n\,Elab\,(b_h :: e)\ a}$$

3.2.2 Check

TODO probably easiest to extend the elaboration checking judgment with the raw terms of the base lang, so everything can move in lock step

$$\frac{H \vdash}{H \vdash \star \, Elab_{\star,l} \, \star}$$

$$\frac{H, f : \Pi x : A.B,\, x : A \vdash m \, Elab_{B,l} \, b}{H \vdash \mathsf{fun} \, f.\, x.\, m \, Elab_{\Pi x:A.B,l} \, \mathsf{fun} \, f.\, x.b}$$

$$\frac{H \vdash m \, Elab \, a_h :: e}{H \vdash m \, Elab_{A,l} \, a_h :: e =_{l,.} A}$$

3.3 Typing

3.3.1 Cast Typing

$$\frac{H \vdash A : \star}{H \vdash A : \overline{\star}} eq - ty - 1$$

$$\frac{H \vdash e : \overline{\star} \quad H \vdash A : \star}{H \vdash e =_{l,o} A : \overline{\star}} eq - ty - 2$$

3.3.2 Head Typing

$$\frac{x : A \in H}{H \vdash x : A} var - ty$$

$$\frac{H \vdash}{H \vdash \star : \star} \star - ty$$

$$\frac{H \vdash A : \star \quad H, x : A \vdash B : \star}{H \vdash \Pi x : A.B \ : \star} \Pi - ty$$

$$\frac{H, f : \Pi x : A.B, x : A \vdash b : B}{H \vdash \mathsf{fun} \, f.\, x.b \ : \ \Pi x : A.B} \Pi - \mathsf{fun} - ty$$

$$\frac{H \vdash b : \Pi x : A.B \quad H \vdash a : A}{H \vdash b \, a \ : \ B \, [x \coloneqq a]} \Pi - app - ty$$

3.3.3 Term Typing

$$\frac{H \vdash}{H \vdash \star : \star}$$

$$\frac{H \vdash A : \star \quad H, x : A \vdash B : \star}{H \vdash \Pi x : A.B \ : \star}$$

$$\frac{H \vdash a : A \quad H \vdash A \equiv A' : \star}{H \vdash a : A'} conv$$

$$\frac{H \vdash e : \overline{\star} \quad H \vdash a_h : B \downarrow}{H \vdash a_h :: e \quad : \quad e \uparrow} apparent$$

3.4 Definitional Equality

$$\frac{H \vdash a \Rrightarrow_* b : A \quad H \vdash a' \Rrightarrow_* b' \; : A \quad b \sim b'}{H \vdash a \equiv a' : A}$$

3.5 Consistent

A relation that equates terms except for source location and observation information

$$\frac{}{\star \sim \star}$$

$$\frac{A \sim A' \quad B \sim B'}{\Pi x : A.B \sim \Pi x : A'.B'}$$

$$\frac{a_h \sim a'_h \quad e \sim e'}{a_h :: e \sim a'_h :: e'}$$

$$\frac{e \sim e' \quad A \sim A'}{e =_{l,o} A \sim e' =_{l',o'} A'}$$

$$\frac{a \sim a'}{\mathsf{fun}\, f.\, x.a \sim \mathsf{fun}\, f.\, x.a'}$$

$$\frac{b \sim b' \quad a \sim a'}{b\, a \sim b'\, a'}$$

3.6 Parallel Reductions

$$\frac{H \vdash a : A}{H \vdash a \Rrightarrow_* a : A}$$

$$\frac{H \vdash a \Rrightarrow_* b : A \quad H \vdash b \Rrightarrow c : A}{H \vdash a \Rrightarrow_* c : A}$$

3.7 Parallel Reduction

3.7.1 Cast Par reduction

$$\frac{H \vdash A \Rrightarrow A' \; : \star}{H \vdash A \Rrightarrow A' \; : \overline{\star}}$$

$$\frac{H \vdash e \Rrightarrow e' : \overline{\star} \quad H \vdash A \Rrightarrow A' \; : \star \quad H \vdash o \Rrightarrow o'}{H \vdash e =_{l,o} A \Rrightarrow e' =_{l,o'} A' \; : \overline{\star}}$$

annoyingly need to support observation reductions, to allow a substitution lemma to simplify the proof

3.7.2 Head Par reduction

$$\frac{H, f : \Pi x : A.B, x : A \vdash b \Rrightarrow b' : B \quad H \vdash a \Rrightarrow a' : A \quad e\, Elim_{\Pi}\, x : e_A.e_B \quad H \vdash e_B \Rrightarrow e'_B : \overline{\star}}{H \vdash (\mathsf{fun}\, f.\, x.b) :: e\, a \Rrightarrow (b' \left[f \coloneqq (\mathsf{fun}\, f.\, x.b'), x \coloneqq a' :: e_A\right] :: e_B \left[x \coloneqq a'\right]) \,:\, B \left[x \coloneqq a\right]} \Pi C \Rrightarrow$$

$$\frac{x : A \in H}{H \vdash x \Rrightarrow x : A}$$

$$\frac{H \vdash}{H \vdash \star \Rrightarrow \star : \star}$$

$$\frac{H \vdash A \Rrightarrow A' : \star \quad H, x : A \vdash B \Rrightarrow B' : \star}{H \vdash \Pi x : A.B \Rrightarrow \Pi x : A'.B' \,:\, \star}$$

$$\frac{H, f : \Pi x : A.B, x : A \vdash b \Rrightarrow b' \,:\, B}{H \vdash \mathsf{fun}\, f.\, x.b \Rrightarrow \mathsf{fun}\, f.\, x.b' \,:\, \Pi x : A.B}$$

$$\frac{H \vdash b \Rrightarrow b' : \Pi x : A.B \quad H \vdash a \Rrightarrow b' : A}{H \vdash b\, a \Rrightarrow b'\, a' \,:\, B \left[x \coloneqq a\right]} \Pi E \Rrightarrow$$

3.7.3 Term Par reduction

$$\frac{H \vdash}{H \vdash \star \Rrightarrow \star : \star}$$

$$\frac{H \vdash A \Rrightarrow A' : \star \quad H, x : A \vdash B \Rrightarrow B' : \star}{H \vdash \Pi x : A.B \Rrightarrow \Pi x : A'.B' \,:\, \star}$$

$$\frac{H \vdash e \Rrightarrow e' : \star \quad H \vdash a \Rrightarrow a' : e \downarrow}{H \vdash a :: e \Rrightarrow a' :: e' \,:\, e \uparrow}$$

3.7.4 Observation Par reduction

$$\frac{H \vdash}{H \vdash . \Rrightarrow .}$$

$$\frac{H \vdash A \Rrightarrow A' : \star \quad H, x : A \vdash B \Rrightarrow B' : \star}{H \vdash \Pi x : A.B \Rrightarrow \Pi x : A'.B' \,:\, \star}$$

$$\frac{H \vdash e \Rrightarrow e' : \star \quad H \vdash a \Rrightarrow a' : e \downarrow}{H \vdash a :: e \Rrightarrow a' :: e' \,:\, e \uparrow}$$

3.8 Dynamic Check

$$\frac{}{\star\, Elim_\star}$$

$$\frac{}{\star :: \star\, Elim_\star}$$

$$\frac{e\, Elim_\star \quad A\, Elim_\star}{e =_{l,o} A\, Elim_\star}$$

$$\frac{}{\Pi x : A.B\, Elim_\Pi\, x : A.B}$$

$$\frac{e\, Elim_\star}{\Pi x : A.B :: e\, Elim_\Pi\, x : A.B}$$

$$\frac{e\, Elim_\Pi\, x : e_A.e_B}{\Pi x : A.B =_{l,o} e\, Elim_\Pi\, x : (A =_{l,o.arg} e_A)\, .e_B\, [x \coloneqq x :: A =_{l,o.arg} A'] =_{l,o.bod[x]} B}$$

$$\frac{e\, Elim_\Pi\, x : e_A.e_B \quad e''\, Elim_\star}{(\Pi x : A.B :: e'') =_{l,o} e\, Elim_\Pi\, x : (A =_{l,o.arg} e_A)\, .e_B\, [x \coloneqq x :: A =_{l,o.arg} A'] =_{l,o.bod[x]} B}$$

4 Call-by-Value Small Step

$$\begin{array}{lcll}
v & ::= & & \star \mid \Pi x : A.B \\
 & & & v_h :: v_{eq} \\
v_h & ::= & \mid & x \\
 & & \mid & \star \\
 & & \mid & \Pi x : A.B \\
 & & \mid & \mathsf{fun}\, f.\, x.a \\
v_{eq} & ::= & & v \\
 & & \mid & v_{eq} =_{l,o} v \\
v_{obs} & ::= & & . \\
 & & \mid & v_{obs}.arg \\
 & & \mid & v_{obs}.bod[v]
\end{array}$$

$$\frac{A \rightsquigarrow A'}{v_{obs}.bod[A] \rightsquigarrow v_{obs}.bod[A']}$$

$$\frac{O \rightsquigarrow O'}{v_{eq} =_{l,O} A \rightsquigarrow v_{eq} =_{l,O'} A}$$

$$\frac{A \rightsquigarrow A'}{v_{eq} =_{l,v_{obs}} A \rightsquigarrow v_{eq} =_{l,v_{obs}} A'}$$

$$\frac{e \rightsquigarrow e'}{e =_{l,o} A \rightsquigarrow e' =_{l,o} A}$$

$$\frac{e \rightsquigarrow e'}{a_h :: e \rightsquigarrow a_h :: e'}$$

$$\frac{a_h \rightsquigarrow a'_h}{a_h :: v_{eq} \rightsquigarrow a_h :: v_{eq}}$$

$$\frac{b \rightsquigarrow b'}{b\, a \rightsquigarrow b'\, a}$$

$$\frac{a \rightsquigarrow a'}{v\, a \rightsquigarrow v\, a'}$$

$$\frac{v_{eq}\, Elim_{\Pi}\, x : e_A . e_B}{(\mathsf{fun}\, f.\, x.b) :: v_{eq}\, v :: v'_{eq} \rightsquigarrow (b\, [f \coloneqq (\mathsf{fun}\, f.\, x.b)\, , x \coloneqq v :: e_A] :: e'_B\, [x \coloneqq v])}$$

(this substitutes non-value casts into values, which is a little awkward but doesn't break anything)