Dependant Types

Initiation to research

Louis Milhaud

Université Paris Saclay

January 9, 2024

Outline

Introduction

History

- 1972: System F, J-Y. Girard
- 1975: Intuitionistic Type Theory, P. Martin-Lof
- 1986: Nuprl System
- 1988: Calculus of Construction, T. Coquand
- 1989: Coq
- 1997: Syntax & Semantics of Dependant Type Theory, M. Hofmann

Definition

A dependant type is a family of types varying on the elements of another type.

Example:

$$Vec_\sigma(M), \quad M : \mathbb{N}$$

with the following objects:

- $nil_\sigma : Vec_\sigma(0)$
- $Cons_\sigma(U, V) : Vec_\sigma(Succ(M))$

with $U : \sigma$ and $V : Vec_\sigma(M)$

Judgments

$\vdash \Gamma \text{ context}$	Γ is a valid context
$\Gamma \vdash \sigma \text{ type}$	σ is a type in context Γ
$\Gamma \vdash M : \sigma$	M is a term of type σ in context Γ
$\vdash \Gamma = \Delta \text{ context}$	Γ and Δ are definitionaly equal contexts
$\Gamma \vdash \sigma = \tau$	σ and τ are def. equal types in context Γ
$\Gamma \vdash M = N : \sigma$	M and N are def. equal terms of σ in Γ

Definitional Equality

Definitional Equality (Per Martin-Lof):

"Definitional equality is intensional equality, or equality of meaning."

This equality defines an equivalence relation:

$$\frac{\vdash \Gamma \text{ context}}{\vdash \Gamma = \Gamma \text{ context}} \qquad \frac{\vdash \Gamma = \Delta \text{ context}}{\vdash \Delta = \Gamma \text{ context}}$$

$$\frac{\vdash \Gamma = \Theta \text{ context} \vdash \Theta = \Delta \text{ context}}{\vdash \Gamma = \Delta \text{ context}}$$

With the same rules for types and terms.

Notions of Equality

Three notions of equality:

- Judgmental equality

$$\frac{\Gamma \vdash \mathbb{N} \text{ type}}{\Gamma \vdash 1 : \mathbb{N}} \qquad \frac{\Gamma \vdash \mathbb{N} \text{ type}}{\Gamma \vdash 1 = \mathit{Suc}(0) : \mathbb{N}}$$

- Typal equality

$$\frac{\Gamma \vdash \mathbb{N} \text{ type}}{\Gamma \vdash 1 : \mathbb{N}} \qquad \frac{\Gamma \vdash \mathbb{N} \text{ type}}{\Gamma \vdash \delta_1 : \mathit{Id}_{\mathbb{N}}(1, \mathit{Suc}(0))}$$

- Propositional equality

$$\frac{\Gamma \vdash \mathbb{N} \text{ type}}{\Gamma \vdash 1 : \mathbb{N}} \qquad \frac{\Gamma \vdash \mathbb{N} \text{ type}}{\Gamma \vdash 1 =_{\mathbb{N}} \mathit{Suc}(0) \text{ true}}$$

Basic rules 1

Rules for context formation:

$$\frac{}{\vdash \diamond \text{ context}} \qquad \frac{\Gamma \vdash \sigma \text{ type}}{\vdash \Gamma, x : \sigma \text{ context}}$$

$$\frac{\Gamma = \Delta \text{ context} \quad \Gamma \vdash \sigma = \tau \text{ type}}{\vdash \Gamma, x : \sigma = \Delta, y : \tau \text{ context}}$$

Variable Rule:

$$\frac{\vdash \Gamma, x : \sigma, \Delta \text{ context}}{\Gamma, x : \sigma, \Delta \vdash x : \sigma}$$

Basic rules 2

Rules for typing and definitional equality:

$$\frac{\Gamma \vdash M : \sigma \quad \vdash \Gamma = \Delta \text{ context} \quad \Gamma \vdash \sigma = \tau \text{ type}}{\Delta \vdash M : \tau}$$

$$\frac{\vdash \Gamma = \Delta \text{ context} \quad \Gamma \vdash \sigma \text{ type}}{\Delta \vdash \sigma \text{ type}}$$

Weakening and substitution Rules:

$$\frac{\Gamma, \Delta \vdash \mathcal{J} \quad \Gamma \vdash \rho \text{ type}}{\Gamma, x : \rho, \Delta \vdash \mathcal{J}}$$

$$\frac{\Gamma, x : \rho, \Delta \vdash \mathcal{J} \quad \Gamma \vdash U : \rho}{\Gamma, \Delta[U/x] \vdash \mathcal{J}[U/x]}$$

Type formers

$\prod$-type

- Type former for the functions with return type depending on the parameter
- Set-theoretic equivalent:
  Cartesian product over a family of sets: $\Pi_{i \in I} B_i$
- preserved by definitional equality

$$\frac{\Gamma \vdash \sigma \text{ type} \quad \Gamma, x : \sigma \vdash \tau \text{ type}}{\Gamma \vdash \Pi x : \sigma . \tau \text{ type}} \mathit{Form}$$

Rules

$$\frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x : \sigma . M^{\tau} : \Pi x : \sigma . \tau} \mathit{Intro}$$

$$\frac{\Gamma \vdash M : \Pi x : \sigma . \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash App_{[x:\sigma]\tau}(M, N) : \tau[N/x]} \mathit{Elim}$$

$$\frac{\Gamma \vdash \lambda x : \sigma . M^{\tau} : \Pi x : \sigma . \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash App_{[x:\sigma]\tau}(\lambda x : \sigma . M^{\tau}, N) = M[N/x] : \tau[N/x]} \mathit{Comp}$$

$\sum$-type

- Type former for pairs with second element type depending on the first element
- Set-theoretic equivalent:

  Disjoint union over a family of sets: $\Sigma_{i \in I} B_i \stackrel{\text{def}}{=} \{(i, b) | i \in I \wedge b \in B_i\}$
- preserved by definitional equality.

$$\frac{\Gamma \vdash \sigma \text{ type} \quad \Gamma, x : \sigma \vdash \tau \text{ type}}{\Gamma \vdash \Sigma x : \sigma . \tau \text{ type}} \mathit{Form}$$

Rules

$$\frac{\Gamma \vdash M : \sigma \quad \Gamma \vdash N : \tau[M/x]}{\Gamma \vdash Pair_{[x:\sigma]\tau}(M, N) : \Sigma x : \sigma.\tau} Intro$$

$$\frac{\begin{array}{c}\Gamma, z : \Sigma x : \sigma.\tau \vdash \rho \text{ type} \\ \Gamma, x : \sigma, y : \tau \vdash H : \rho[Pair_{[x:\sigma]\tau}(x, y)/z] \\ \Gamma \vdash M : \Sigma x : \sigma.\tau\end{array}}{\Gamma \vdash \boldsymbol{R}^{\Sigma}_{[z:\Sigma x:\sigma.\tau]\rho}([x : \sigma.y : \tau]H, M) : \rho[M/z]} Elim$$

How to use the dependant sum

$$\frac{\Gamma \vdash \boldsymbol{R}^{\Sigma}_{[z:\Sigma x:\sigma.\tau]\rho}([x:\sigma, y:\tau]H, Pair_{[x:\sigma]\tau}(M,N)) : \rho[Pair_{[x:\sigma]\tau}(M,N)/z]}{\Gamma \vdash \boldsymbol{R}^{\Sigma}_{[z:\Sigma x:\sigma.\tau]\rho}([x:\sigma, y:\tau]H, Pair_{[x:\sigma]\tau}(M,N)) \\ = H[M/x, N/y] : \rho[Pair_{[x:\sigma]\tau}(M,N)/z].} \; Comp$$

Projection definitions:

$$\pi_1 \stackrel{\text{def}}{=} \boldsymbol{R}^{\Sigma}_{[z:\Sigma x:\sigma.\tau]\rho}([x:\sigma, y:\tau]x, M) : \sigma$$

$$\pi_2 \stackrel{\text{def}}{=} \boldsymbol{R}^{\Sigma}_{[z:\Sigma x:\sigma.\tau]\tau[z.1/x]}([x:\sigma, y:\tau]y, M) : \tau[M.1]$$

Naturals

Naturals are as always inductively defined, thus we have the two following objects:

$$0 \text{ and } Suc(M) \text{ with } M : \mathbb{N}$$

We then have rules for each object:

$$\frac{\vdash \Gamma \text{ context}}{\Gamma \vdash \mathbb{N} \text{ type}} \textit{Form}$$

$$\frac{\vdash \Gamma \text{ context}}{\Gamma \vdash 0 : \mathbb{N}} \qquad \frac{\Gamma \vdash M : \mathbb{N}}{\Gamma \vdash Suc(M) : \mathbb{N}} \textit{Intro}$$

Rules

$$\frac{\begin{array}{c} \Gamma, n : \mathbb{N} \vdash \sigma \text{ type} \\ \Gamma \vdash H_z : \sigma[0/n] \\ \Gamma, n : \mathbb{N}, x : \sigma \vdash H_s : \sigma[Suc(n)/n] \\ \Gamma \vdash M : \mathbb{N} \end{array}}{\Gamma \vdash \boldsymbol{R}^{\mathbb{N}}_{[n:\mathbb{N}]\sigma}(H_z, [n : \mathbb{N}, x : \sigma]H_s, M) : \sigma[M/n]} Elim$$

Naturals: usage

Computation rules:

$$\Gamma \vdash \boldsymbol{R}^{\mathbb{N}}_{[n:\mathbb{N}]\sigma}(H_z, [n : \mathbb{N}, x : \sigma]H_s, 0) = H_z : \sigma[0/n] \tag{C-0}$$

$$\Gamma \vdash \boldsymbol{R}^{\mathbb{N}}_{[n:\mathbb{N}]\sigma}(H_z, [n : \mathbb{N}, x : \sigma]H_s, Suc(M)) \tag{C-S}$$

$$= H_s[M/n, \boldsymbol{R}^{\mathbb{N}}_{[n:\mathbb{N}]\sigma}(H_z, [n : \mathbb{N}, x : \sigma]H_s, M)/x] : \sigma[Suc(M)/n]$$

Then we can define operations like addition:

$$M + N \stackrel{\text{def}}{=} \boldsymbol{R}^{\mathbb{N}}_{[n:\mathbb{N}]\sigma}(N, [n : \mathbb{N}, x : \mathbb{N}]Suc(x), M) : \mathbb{N}$$

Identity types

- Type former for typal equality
- $\mathit{Refl}_\sigma(M) : \mathit{Id}_\sigma(M, M)$ with $M : \sigma$

$$\frac{\Gamma \vdash M : \sigma \quad \Gamma \vdash N : \sigma}{\Gamma \vdash \mathit{Id}_\sigma(M, N)\ \mathrm{type}}\mathit{Form}$$

$$\frac{\Gamma \vdash M : \sigma}{\Gamma \vdash \mathit{Refl}_\sigma(M) : \mathit{Id}_\sigma(M, M)}\mathit{Intro}$$

Rules

$$\frac{\begin{array}{c}\Gamma \vdash \sigma \text{ type} \quad \Gamma \vdash M : \sigma \quad \Gamma \vdash N : \sigma \\ \Gamma, x : \sigma, y : \sigma, p : Id_\sigma(x, y) \vdash \tau \text{ type} \\ \Gamma, z : \sigma \vdash H : \tau[z/x, z/y, Refl_\sigma(z)/p] \\ \Gamma \vdash P : Id_\sigma(M, N)\end{array}}{\Gamma \vdash \boldsymbol{R}^{Id}_{[x:\sigma, y:\sigma, p:Id_\sigma(x,y)]\tau}([z : \sigma]H, M, N, P) : \tau[M/x, N/y, P/p]} Elim$$

Intensional and Extensional type theory

- type theory + identity types = intensional type theory
- equality reflection:

$$\frac{\Gamma \vdash P : Id_\sigma(M, N)}{\Gamma \vdash M = N : \sigma}$$

- type theory + identity types + reflection rule = extensional type theory
- Judgmental equality and typing undecidable

Universes

- Universes are type formers containing codes for types
- We have the two basic rules:

$$\frac{\vdash \Gamma \text{ context}}{\Gamma \vdash U \text{ type}} \qquad \frac{\Gamma \vdash M : U}{\Gamma \vdash El(M) \text{ type}}$$

- M is a code of the universe U
- $El(M)$ is the type associated to this code
- Add type formers to our Universes
- Rules to maintain closure

Adding dependant type $\hat{\Pi}$

$$\frac{\Gamma \vdash S : U \quad \Gamma, s : El(S) \vdash T : U}{\Gamma \vdash \hat{\Pi}(S, [s : El(S)]T) : U}\,Form$$

$$\frac{\Gamma \vdash \hat{\Pi}(S, [s : El(S)]T) : U}{\Gamma \vdash El(\hat{\Pi}(S, [s : El(S)]T)) = \Pi s : El(S).El(T) \ \mathrm{type}}$$

Impredicative quantification

- $\forall x : \sigma.T : U$
- Important for universe closure
- σ arbitrarily big
- $polyone = \forall c : U.\forall s : El(c).s$

$$\frac{\Gamma \vdash \sigma \text{ type} \quad \Gamma, x : \sigma \vdash T : U}{\Gamma \vdash \forall x : \sigma.T : U}$$

Impredicative quantification - Rules

$$\frac{\Gamma, x : \sigma \vdash M : El(T)}{\Gamma \vdash \hat{\lambda} x : \sigma . M^{El(T)} : El(\forall x : \sigma . T)} \, Intro$$

$$\frac{\Gamma \vdash M : El(\forall x : \sigma . T) \quad \Gamma \vdash N : \sigma}{\Gamma \vdash \hat{App}_{[x:\sigma]El(T)}(M, N) : El(T[N/x])} \, Elim$$

$$\frac{\Gamma \vdash \hat{App}_{[x:\sigma]El(T)}(\hat{\lambda} x : \sigma . M^{El(T)}, N) : El(T[N/x])}{\Gamma \vdash \hat{App}_{[x:\sigma]El(T)}(\hat{\lambda} x : \sigma . M^{El(T)}, N) = M[N/x] : El(T[N/x])} \, Comp$$

Categoretical Semantics

Context morphisms

Let Γ and $\Delta \stackrel{\text{def}}{=} x_1 : \sigma_1, ..., x_n : \sigma_n$ be valid contexts.
$f \stackrel{\text{def}}{=} (M_1, ..., M_n)$ is a context morphism (we write $\Gamma \vdash f \implies \Delta$)
when the following n judgements hold:

$$
\begin{aligned}
&\Gamma \vdash M_1 : \sigma_1 \\
&\Gamma \vdash M_2 : \sigma_2[M_1/x_1] \\
&\dots \\
&\Gamma \vdash M_n : \sigma_n[M_1/x_1][M_2/x_2]\dots[M_{n-1}/x_{n-1}]
\end{aligned}
$$

Categoretical Semantics

Our basic Syntax contains now 4 objects

- contexts
- contexts morphisms
- types in Ty
- terms in Tm

And we define the corresponding Semantic:

- $\mathcal{C}$ a category of contexts and context morphisms
- for $\Gamma \in \mathcal{C}$ a collection $Ty_{\mathcal{C}}(\Gamma)$ of semantic types
- for $\Gamma \in \mathcal{C}$ and $\sigma \in Ty_{\mathcal{C}}$ a collection $Tm_{\mathcal{C}}(\Gamma, \sigma)$ of semantic terms

Category $\mathcal{F}am$

The category $\mathcal{F}am$ of families of sets has:

- as objects pairs $A = (A^0, A^1)$
- as arrows f between A and B a pair (f^0, f^1)

We also define the functor $\mathcal{F} : \mathcal{C}^{op} \to \mathcal{F}am$ such that:

$$\mathcal{F}(\Gamma) = (Ty_{\mathcal{C}}(\Gamma), (Tm_{\mathcal{C}}(\Gamma, \sigma))_{\sigma \in Ty_{\mathcal{C}}(\Gamma)})$$

Category with families

We can now define a category with families with:

- a category $\mathcal{C}$ with terminal object
- a functor $\mathcal{F} = (Ty, Tm) : \mathcal{C}^{op} \rightarrow \mathcal{F}am$
- a comprehension for each $\Gamma \in \mathcal{C}$ and $\sigma \in Ty_{\mathcal{C}}(\Gamma)$

To have a complete semantic, we would have to define the terms and substitution...