Types are weak omega-groupoids, in Coq

A. Lafont[1], T. Hirschowitz[2], N. Tabareau[3]

[1]IMT Atlantique, Nantes, France,[2]CNRS, Université Savoie Mont-Blanc, France,[3]Inria, Nantes, France

TYPES, 2018

Introduction

- In a Martin-Löf Type theory, given two terms t, u of the same type A, there is an associated identity type $t =_A u$.

- The eliminator J for this identity type allows to compose equalities, invert them, etc.

- We can consider the (two-dimensional) identity type between two proofs of equalities $p =_{t =_A u} q$, or even higher-dimensional identities ($\alpha =_{p =_{t =_A u} q} \beta$ and so on).

- J induces more operations mixing different dimensions (whiskerings for example). These operations give the base type A and its iterated identity types the structure of a weak omega groupoid.

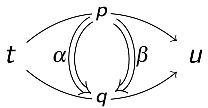
Introduction

- In a Martin-Löf Type theory, given two terms t, u of the same type A, there is an associated identity type $t =_A u$.
- The eliminator **J** for this identity type allows to compose equalities, invert them, etc.
- We can consider the (two-dimensional) identity type between two proofs of equalities $p =_{t=_A u} q$, or even higher-dimensional identities ($\alpha =_{p=_{t=_A u} q} \beta$ and so on).

- **J** induces more operations mixing different dimensions (whiskerings for example). These operations give the base type A and its iterated identity types the structure of a weak omega groupoid.

Introduction

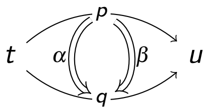
- In a Martin-Löf Type theory, given two terms t, u of the same type A, there is an associated identity type $t =_A u$.
- The eliminator **J** for this identity type allows to compose equalities, invert them, etc.
- We can consider the (two-dimensional) identity type between two proofs of equalities $p =_{t =_A u} q$, or even higher-dimensional identities ($\alpha =_{p =_{t =_A u} q} \beta$ and so on).



- **J** induces more operations mixing different dimensions (whiskerings for example). These operations give the base type A and its iterated identity types the structure of a weak omega groupoid.

Introduction

- In a Martin-Löf Type theory, given two terms t, u of the same type A, there is an associated identity type $t =_A u$.
- The eliminator **J** for this identity type allows to compose equalities, invert them, etc.
- We can consider the (two-dimensional) identity type between two proofs of equalities $p =_{t =_A u} q$, or even higher-dimensional identities ($\alpha =_{p =_{t =_A u} q} \beta$ and so on).



- **J** induces more operations mixing different dimensions (whiskerings for example). These operations give the base type A and its iterated identity types the structure of a weak omega groupoid.

Types are weak omega groupoids

- The proof on paper first appears in *Lumsdaine* 2010, *van den Berg and Garner* 2011.
- This proof about Type Theory has not yet been formalized internally in Type Theory

Main Coq formalized statment:

Any (fibrant) type (of Coq) has a structure of weak omega groupoid induced by its iterated identity types

- in a **two-level type system**
- through the encoding of **Brunerie Type Theory** as an **extrinsic syntax**

Types are weak omega groupoids

- The proof on paper first appears in *Lumsdaine* 2010, *van den Berg and Garner* 2011.
- This proof about Type Theory has not yet been formalized internally in Type Theory

Main Coq formalized statment:

Any (fibrant) type (of Coq) has a structure of weak omega groupoid induced by its iterated identity types

- in a **two-level type system**
- through the encoding of **Brunerie Type Theory** as an **extrinsic syntax**

Outline

A type theoretical definition of weak omega-groupoids

- Globular sets
- Brunerie Type Theory

Formalization in Coq

- Intrinsic vs Extrinsic
- Two-level type system

Perspectives

Outline

A type theoretical definition of weak omega-groupoids

- Globular sets
- Brunerie Type Theory

Formalization in Coq

- Intrinsic vs Extrinsic
- Two-level type system

Perspectives

Globular sets as models of a type theory.

A **model** of a type theory consists in interpreting

- closed types (and contexts) as sets
- closed terms as elements of their type

- open types as families over the context
- open terms as sections of their open types

$$[\![\Gamma \vdash A\ \mathit{Type}]\!] : [\![\Gamma]\!] \to \mathit{Set}$$

$$[\![\Gamma \vdash t : A]\!] : \prod(\gamma : [\![\Gamma]\!]), [\![\Gamma \vdash A\ \mathit{Type}]\!]\gamma$$

- + some compatibility relations

$$[\![\Gamma, x : A]\!] = \sum(\gamma : [\![\Gamma]\!]), [\![\Gamma \vdash A\ \mathit{Type}]\!]\gamma$$

...

In short, a model is a CwF morphism from the type theory to the standard CwF structure on the *Set* category.

Globular sets as models of a type theory.

Globular Type Theory : a Type Theory with a constant type $\star$ and identity types, but without any constructor and eliminator:

$$\frac{\Gamma \vdash}{\Gamma \vdash \star\ Type} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : A}{\Gamma \vdash t =_A u\ Type}$$

A model of this type theory consists of

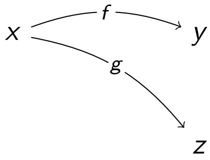
- a set $[\![\star]\!]$
- for each $x, y \in [\![\star]\!]$, a set $[\![x =_A y]\!]$
- for each $f, g \in [\![x =_A y]\!]$, a set $[\![f =_A g]\!]$
- ...

Models of this type theory correspond to globular sets

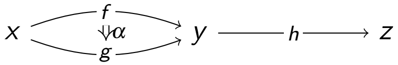
Globular sets as models of a type theory.

Contexts correspond to finite globular sets:

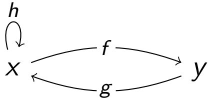
- $x : \star, y : \star, z : \star, f : x =_\star y, g : x =_\star z$



- $x : \star, y : \star, z : \star, f : x =_\star y, g : x =_\star y, \alpha : f =_{x =_\star y} g, h : y =_\star z$



- $x : \star, y : \star, z : \star, f : x =_\star y, g : y =_\star x, h : x =_\star x$



Outline

Brunerie Type Theory.

Brunerie Type Theory : an enrichment of the previous globular Type Theory such that models are equipped with expected coherences (composition, reflexivity, ...).

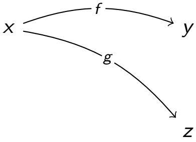
- Same type constructors (a constant $\star$ and identity types)
- For each type in a **contractible context**, a term that inhabit it:

$$\frac{\Gamma \vdash_c \quad \Gamma \vdash A\ Type}{\Gamma \vdash coh : A}$$

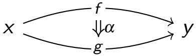
Brunerie Type Theory.

- **contractible contexts** characterized inductively:

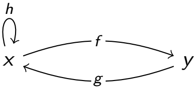
$$\frac{}{x : \star \vdash_c} \qquad \frac{\Gamma \vdash_c \quad y : A \in \Gamma}{\Gamma, z : A, h : y =_A z \vdash_c}$$



$$\frac{\dfrac{\dfrac{}{x : \star \vdash_c} \qquad x : \star \in x : \star}{x : \star, y : \star, f : x =_\star y \vdash_c} \qquad x : \star \in x : \star, y : \star, f : x =_\star y}{x : \star, y : \star, f : x =_\star y, z : \star, g : x =_\star z \vdash_c}$$



$$\frac{\dfrac{\dfrac{}{x : \star \vdash_c} \qquad x : \star \in x : \star}{x : \star, y : \star, f : x =_\star y \vdash_c} \qquad f : x =_\star y \in x, y : \star, f : x =_\star y}{x : \star, y : \star, f : x =_\star y, g : x =_\star y, \alpha : f =_{x =_\star y} g \vdash_c}$$



is not contractible

Brunerie Type Theory.

Contexts and contractible contexts

$$\emptyset \vdash \qquad \frac{\Gamma \vdash A\ Type}{\Gamma, x : A \vdash}$$

$$\frac{}{x : \star \vdash_c} \qquad \frac{y : A \in \Gamma \quad \Gamma \vdash_c}{\Gamma, x : A, f : y =_A x \vdash_c}$$

Types

$$\frac{\Gamma \vdash}{\Gamma \vdash \star\ Type} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : A}{\Gamma \vdash t =_A u\ Type}$$

Terms

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{\Gamma \vdash \sigma : \Delta \qquad \Delta \vdash A\ Type \qquad \Delta \vdash_c}{\Gamma \vdash coh_{\Delta,A,\sigma} : A[\sigma]}$$

Substitutions

$$\frac{}{\Gamma \vdash () : \emptyset} \qquad \frac{\Gamma \vdash \sigma : \Delta \quad \Delta \vdash A \quad \Gamma \vdash t : A[\sigma]}{\Gamma \vdash (\sigma, x \mapsto t) : \Delta, x : A}$$

$$\star[\sigma] := \star \qquad x[\sigma] := \sigma(x) \quad \text{(variable)}$$

$$(t =_A u)[\sigma] := t[\sigma] =_{A[\sigma]} u[\sigma] \quad coh_{\Delta,A,\delta}[\sigma] := coh_{\Delta,A,\sigma\circ\delta}$$

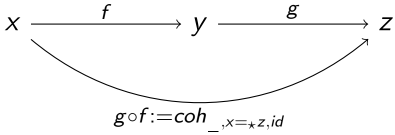
Examples of derivations

Identities

$$\frac{x : \star \vdash_c}{x : * \vdash_c coh_{x:\star, x=_\star x, id} : x =_\star x}$$

Composition

$$\frac{x : \star, y : \star, f : x =_\star y, z : \star, g : y =_\star z \vdash_c}{x : \star, y : \star, f : x =_\star y, z : \star, g : y =_\star z \vdash_c coh_{\_, x=_\star z, id} : x =_\star z}$$



Outline

Brunerie Type Theory.

Contexts and contractible contexts

$$\emptyset \vdash \qquad \frac{\Gamma \vdash A\ \mathit{Type}}{\Gamma, x : A \vdash}$$

$$\frac{}{x : \star \vdash_c} \qquad \frac{\Gamma \vdash_c \quad y : A \in \Gamma}{\Gamma, x : A, f : x =_A y \vdash_c}$$

Types

$$\frac{\Gamma \vdash}{\Gamma \vdash \star\ \mathit{Type}} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : A}{\Gamma \vdash t =_A u\ \mathit{Type}}$$

Terms

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{\Gamma \vdash \sigma : \Delta \qquad \Delta \vdash A\ \mathit{Type} \qquad \Delta \vdash_c}{\Gamma \vdash coh_{\Delta,A,\sigma} : A[\sigma]}$$

Substitutions

$$\frac{}{\Gamma \vdash () : \emptyset} \qquad \frac{\Gamma \vdash \sigma : \Delta \quad \Delta \vdash A \quad \Gamma \vdash t : A[\sigma]}{\Gamma \vdash (\sigma, x \mapsto t) : \Delta, x : A}$$

$$\star[\sigma] := \star \qquad x[\sigma] := \sigma(x) \quad \text{(variable)}$$

$$(t =_A u)[\sigma] := t[\sigma] =_{A[\sigma]} u[\sigma] \qquad coh_{\Delta,A,\delta}[\sigma] := coh_{\Delta,A,\sigma\circ\delta}$$

Brunerie Type Theory.

- There is no conversion: no need to quotient the syntax by a convertibility relation
- The typing of coherence terms require to be able to compute substitution:

$$\frac{\Gamma \vdash \sigma : \Delta \qquad \Delta \vdash A\ Type \qquad \Delta \vdash_c}{\Gamma \vdash coh_{\Delta,A,\sigma} : A[\sigma]}$$

Formalization : Intrinsic syntax or Extrinsic syntax ?

Brunerie Type Theory.

- There is no conversion: no need to quotient the syntax by a convertibility relation
- The typing of coherence terms require to be able to compute substitution:

$$\frac{\Gamma \vdash \sigma : \Delta \qquad \Delta \vdash A\ Type \qquad \Delta \vdash_c}{\Gamma \vdash coh_{\Delta,A,\sigma} : A[\sigma]}$$

Formalization : Intrinsic syntax or Extrinsic syntax ?

Intrinsic syntax for Brunerie Type Theory

Intrinsic syntax: define the typed syntax directly, without mentionning untyped syntax.

Inductive-Inductive-Recursive datatype:

$$
\begin{aligned}
&\mathit{Inductive}\ \mathit{Con} : \mathscr{U} \\
&\qquad \mathit{with}\ \mathit{Ty} : \mathit{Con} \to \mathscr{U} \\
&\qquad \mathit{with}\ \mathit{Tm} : \forall(\Gamma : \mathit{Con}), \mathit{Ty}\,\Gamma \to \mathscr{U} := \ldots \\
&\qquad\quad |\mathit{coh} : \forall \Gamma\,\Delta\,(\sigma : \mathit{Sub}\,\Gamma\,\Delta)(A : \mathit{Ty}\,\Delta), \mathit{isContr}\,\Delta \to \mathit{Tm}\,\Gamma(\mathit{subTy}\,\sigma\,A) \\
&\qquad\qquad \ldots\ldots \\
&\mathit{with}\ \mathit{fix}\ \mathit{subTy}\,(\Gamma\,\Delta : \mathit{Con})(\sigma : \mathit{Sub}\,\Gamma\,\Delta)(A : \mathit{Ty}\,\Delta) : \mathit{Ty}\,\Gamma := \ldots
\end{aligned}
$$

See Agda formalization by Nuo Li (*Some constructions on ω-groupoids*, LFMTP 2014, Altenkirch & Li).

Extrinsic syntax for Brunerie Type Theory

Our Coq formalization follows a different path (Coq does not support Inductive-Inductive-Recursive datatypes):

- **Extrinsic syntax**: define the untyped syntax and then the well-typed judgements

1. First, define the untyped syntax:

$$\begin{aligned} \textit{Inductive Con} : \mathscr{U} &:= \ldots \\ \textit{with Ty} : \mathscr{U} &:= \ldots \\ \textit{with Tm} : \mathscr{U} &:= \ldots \\ &\ldots \end{aligned}$$

2. then, define $\textit{fix subTy}\, \sigma\, A : Ty := \ldots$ on the untyped syntax, and
3. finally define the following well-typed judgements:

$$\begin{aligned} \textit{Inductive Conw} &: Con \to \mathscr{U} \\ \textit{with Tyw} &: Con \to Ty \to \mathscr{U} \\ \textit{with Tmw} &: Con \to Ty \to Tm \to \mathscr{U} \end{aligned}$$

Outline

Deriving the (non dependent) recursor

We formalize the proof that any type A with its iterated identity types is a weak omega groupoid (i.e. gives a model of Brunerie Type Theory):

1. by induction on the syntax
2. (roughly) by repeated applications of **J**

It requires to derive the non dependent recursor .

Strategy

1. define the relation $\sim$ specifying the recursor:

$$x \sim y \text{ iff } rec\, x = y$$

2. show that it is atually a functional relation:

$$\forall x \exists! y \text{ s.t. } x \sim y$$

3. extract from the previous step the only element y that is related to the argument x. Define:

$$rec\, x := y$$

Deriving the (non dependent) recursor

We formalize the proof that any type A with its iterated identity types is a weak omega groupoid (i.e. gives a model of Brunerie Type Theory):

1. by induction on the syntax
2. (roughly) by repeated applications of **J**

It requires to derive the non dependent recursor .

Strategy

1. define the relation $\sim$ specifying the recursor:
$$x \sim y \text{ iff } rec\, x = y$$
2. show that it is atually a functional relation:
$$\forall x \exists ! y \text{ s.t. } x \sim y$$
3. extract from the previous step the only element y that is related to the argument x. Define:
$$rec\, x := y$$

Deriving the (non dependent) recursor

We formalize the proof that any type A with its iterated identity types is a weak omega groupoid (i.e. gives a model of Brunerie Type Theory):

1. by induction on the syntax
2. (roughly) by repeated applications of **J**

It requires to derive the non dependent recursor .

Strategy

1. define the relation $\sim$ specifying the recursor:

$$x \sim y \text{ iff } rec\, x = y$$

2. show that it is atually a functional relation:

$$\forall x \exists ! y \text{ s.t. } x \sim y$$

3. extract from the previous step the only element y that is related to the argument x. Define:

$$rec\, x := y$$

Two level type system

I managed to perform the second step only for set-truncated arguments (or by assuming Uniqueness of Identity Proofs UIP).

- Assuming UIP makes the omega groupoid structure of a type trivial
- Solution: a type theory with two equalities (the so-called **two level type system**)
  - a "strict" equality with UIP (and funext)
  - a "homotopical" equality without UIP, used to give the (non trivial) structure of weak omega groupoid to a (fibrant) type

Two level type system

I managed to perform the second step only for set-truncated arguments (or by assuming Uniqueness of Identity Proofs UIP).

- Assuming UIP makes the omega groupoid structure of a type trivial
- Solution: a type theory with two equalities (the so-called **two level type system**)
  - a "strict" equality with UIP (and funext)
  - a "homotopical" equality without UIP, used to give the (non trivial) structure of weak omega groupoid to a (fibrant) type

Two level type system

I managed to perform the second step only for set-truncated arguments (or by assuming Uniqueness of Identity Proofs UIP).

- Assuming UIP makes the omega groupoid structure of a type trivial
- Solution: a type theory with two equalities (the so-called **two level type system**)
    - a "strict" equality with UIP (and funext)
    - a "homotopical" equality without UIP, used to give the (non trivial) structure of weak omega groupoid to a (fibrant) type

Outline

Perspectives and work in progress

Remark: Brunerie Type Theory can be modified to yield a definition of weak omega categories:

A type theoretical definition of weak omega categories, E. Finster & S. Mimram, LICS 2017

- WIP: show unicity of the recursor of Brunerie Type Theory and derive the dependent eliminator
- Does this technique work for other Inductive-Inductive-Recursive datatypes ?

The end

Thank you for your attention!