Dimensional Information in a Type Theory with a Definitionally Proof Irrelevant Layer

Julien Marquet-Wagner [1] Sophie d'Espalungue [2]

[1]École normale supérieure – PSL, Paris, France
julien.marquet@ens.fr

[2]IRIF, CNRS, Paris, France
despalungue@irif.fr

June 5, 2026

Our Work

[†] https://github.com/thejohncrafter/SSTT26/

[‡] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

[§] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". 2023.

Our Work

An implementation[†] of a type theory in Agda.

[†] https://github.com/thejohncrafter/SSTT26/

[‡] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

[§] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". 2023.

Our Work

An implementation[†] of a type theory in Agda.

Featuring **Definitional Proof-Irrelevance**

Let P be a proposition and $p, q : P$ be proofs.
We have $p \equiv q$.

[†] https://github.com/thejohncrafter/SSTT26/

[‡] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

[§] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". 2023.

Our Work

An implementation[†] of a type theory in Agda.

Featuring **Definitional Proof-Irrelevance**

Let P be a proposition and $p, q : P$ be proofs.
We have $p \equiv q$.

Methods

[†] https://github.com/thejohncrafter/SSTT26/
[‡] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.
[§] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". 2023.

Our Work

An implementation[†] of a type theory in Agda.

Featuring **Definitional Proof-Irrelevance**

Let P be a proposition and $p, q : P$ be proofs.
We have $p \equiv q$.

Methods

- A methodology for mechanizations of dependent type theories[‡].

[†] https://github.com/thejohncrafter/SSTT26/

[‡] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

[§] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". 2023.

Our Work

An implementation[†] of a type theory in Agda.

Featuring **Definitional Proof-Irrelevance**

Let P be a proposition and $p, q : P$ be proofs.
We have $p \equiv q$.

Methods

- A methodology for mechanizations of dependent type theories[‡].
- A framework for building n-categories inductively from their $(n-1)$-categories of morphisms[§].

[†] https://github.com/thejohncrafter/SSTT26/

[‡] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

[§] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". 2023.

Our Work

An implementation[†] of a type theory in Agda.

Featuring **Definitional Proof-Irrelevance**

Let P be a proposition and $p, q : P$ be proofs.
We have $p \equiv q$.

Methods

- A methodology for mechanizations of dependent type theories[‡].
- A framework for building n-categories inductively from their $(n-1)$-categories of morphisms[§].
- **Dimensional layers to control (ir)relevance.**

[†] https://github.com/thejohncrafter/SSTT26/

[‡] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

[§] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". 2023.

This Talk

[†]Gaëtan Gilbert et al. “Definitional proof-irrelevance without K”. 2019.

This Talk

This talk is about how thinking in terms of categorical dimension is useful for implementers.

[†]Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

This Talk

This talk is about how thinking in terms of categorical dimension is useful for implementers.

I am convinced it is, because this point of view got me a simple solution to a difficult engineering problem.

[†]Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

This Talk

This talk is about how thinking in terms of categorical dimension is useful for implementers.

I am convinced it is, because this point of view got me a simple solution to a difficult engineering problem.

1. Reference implementations: following Gilbert *et. al.*[†].

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

This Talk

This talk is about how thinking in terms of categorical dimension is useful for implementers.

I am convinced it is, because this point of view got me a simple solution to a difficult engineering problem.

1. Reference implementations: following Gilbert *et. al.*[†].
2. My approach to implementing type theory.

[†]Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

This Talk

This talk is about how thinking in terms of categorical dimension is useful for implementers.

I am convinced it is, because this point of view got me a simple solution to a difficult engineering problem.

1. Reference implementations: following Gilbert *et. al.*[†].
2. My approach to implementing type theory.
3. How the two don't match.

[†]Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

This Talk

This talk is about how thinking in terms of categorical dimension is useful for implementers.

I am convinced it is, because this point of view got me a simple solution to a difficult engineering problem.

1. Reference implementations: following Gilbert *et. al.*[†].
2. My approach to implementing type theory.
3. How the two don't match.
4. How to resolve this tension and the geometrical nature of the solution.

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

If You Only Remember One Slide

If You Only Remember One Slide

low-dimensional category theory

Sets	dim. 0
Propositions	dim.-1

This dimension determines the status of equality:

For Propositions

P a proposition and $p, q : P$ proofs
We have $p \equiv q$ meaning p and q are definitionally equal.

For Sets

X a set and $x, y : X$ points
There is a proposition denoted $x = y$, of which proofs witness the equality of x and y.

Definitional Proof-Irrelevance without K

[†]Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.
[‡]loc. cit. §3.2
[§]loc. cit. Figure 1

Definitional Proof-Irrelevance without K

Gilbert, *Cockx*, *Sozeau*, *and Tabareau* describe[†] the implementation of definitional proof-irrelevance in Rocq and Agda.

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.
[‡] loc. cit. §3.2
[§] loc. cit. Figure 1

Definitional Proof-Irrelevance without K

Gilbert, Cockx, Sozeau, and Tabareau describe[†] the implementation of definitional proof-irrelevance in Rocq and Agda.

The statement is as follows[‡]

$$\frac{\Gamma \vdash A : \textsc{sProp} \quad \Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x \equiv y : A}$$

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

[‡] loc. cit. §3.2

[§] loc. cit. Figure 1

Definitional Proof-Irrelevance without K

Gilbert, Cockx, Sozeau, and Tabareau describe[†] the implementation of definitional proof-irrelevance in Rocq and Agda.

The statement is as follows[‡]

$$\frac{\Gamma \vdash A : \textsc{sProp} \quad \Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x \equiv y : A}$$

Notice how A can feely appear on the left and on the right of the colon. In their definition of the type theory[§], types and terms are defined at the same time with the following BNF

$$\begin{array}{rcl} A, B, M, N & ::= & \textsc{Type}_i \mid x \mid \lambda x : A.M \mid \Pi x : A.B \mid \Sigma x : A.B \mid \pi_1\, M \mid \pi_2\, M \mid (M, N) \\ \Gamma, \Delta & ::= & . \mid \Gamma, x : A \end{array}$$

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

[‡] loc. cit. §3.2

[§] loc. cit. Figure 1

Definitional Proof-Irrelevance without K

Gilbert, Cockx, Sozeau, and Tabareau describe[†] the implementation of definitional proof-irrelevance in Rocq and Agda.

The statement is as follows[‡]

$$\frac{\Gamma \vdash A : \textsc{sProp} \qquad \Gamma \vdash x : A \qquad \Gamma \vdash y : A}{\Gamma \vdash x \equiv y : A}$$

Notice how A can feely appear on the left and on the right of the colon. In their definition of the type theory[§], types and terms are defined at the same time with the following BNF

$$\begin{array}{rcl} A, B, M, N & ::= & \textsc{Type}_i \mid x \mid \lambda x : A.M \mid \Pi x : A.B \mid \Sigma x : A.B \mid \pi_1\, M \mid \pi_2\, M \mid (M, N) \\ \Gamma, \Delta & ::= & . \mid \Gamma, x : A \end{array}$$

Terminology. Types and terms are merged in *expressions*.

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

[‡] loc. cit. §3.2

[§] loc. cit. Figure 1

Types, Terms, and Expressions

Types, Terms, and Expressions

Working with expressions (no distinction between types and terms) makes the left and the right of the colon " : " uniform.

Types, Terms, and Expressions

Working with expressions (no distinction between types and terms) makes the left and the right of the colon " : " uniform.

Universes control the (ir)relevance of the data:

$$\frac{\ldots \quad \Gamma \vdash A : \textsc{Type} \quad \ldots}{A \text{ is proof-relevant}} \qquad \frac{\ldots \quad \Gamma \vdash A : \textsc{sProp} \quad \ldots}{A \text{ is proof-irrelevant}}$$

Mechanizing Type Theory: A Methodology

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.
2. Typing derivations.

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.
2. Typing derivations.

Data Type *Typing Derivation* *Meaning*

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.
2. Typing derivations.

Data Type	*Typing Derivation*	*Meaning*
$\Gamma : \mathsf{Ctx}$	$\Gamma \vdash$	Γ is a well-formed context

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.
2. Typing derivations.

Data Type	*Typing Derivation*	*Meaning*
Γ : Ctx	$\Gamma \vdash$	Γ is a well-formed context
v : Var Γ	*(none)*	v is a variable in Γ

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.
2. Typing derivations.

Data Type	*Typing Derivation*	*Meaning*
$\Gamma : \mathsf{Ctx}$	$\Gamma \vdash$	Γ is a well-formed context
$v : \mathsf{Var}\ \Gamma$	*(none)*	v is a variable in Γ
$A : \mathsf{Typ}\ \Gamma$	$\Gamma \vdash A$	A is well-formed in context Γ

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.
2. Typing derivations.

Data Type	*Typing Derivation*	*Meaning*
$\Gamma : \mathsf{Ctx}$	$\Gamma \vdash$	Γ is a well-formed context
$v : \mathsf{Var}\ \Gamma$	*(none)*	v is a variable in Γ
$A : \mathsf{Typ}\ \Gamma$	$\Gamma \vdash A$	A is well-formed in context Γ
$t : \mathsf{Trm}\ \Gamma$	$\Gamma \vdash t :: A$	t has type A in context Γ

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Mechanizing Type Theory: A Methodology

I am working on an *intrinsically scoped* and *extrinsically typed* presentation of type theory[†]. The formal construction happens in two stages.

1. Well-scoped, untyped syntax.
2. Typing derivations.

Data Type	*Typing Derivation*	*Meaning*
$\Gamma : \mathsf{Ctx}$	$\Gamma \vdash$	Γ is a well-formed context
$v : \mathsf{Var}\ \Gamma$	*(none)*	v is a variable in Γ
$A : \mathsf{Typ}\ \Gamma$	$\Gamma \vdash A$	A is well-formed in context Γ
$t : \mathsf{Trm}\ \Gamma$	$\Gamma \vdash t :: A$	t has type A in context Γ

Specificity of my approach: notice how variables, types and terms are required to live in a context.

[†] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". 2026.

Russell/Tarski Universes

[†]Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

Russell/Tarski Universes

In my work, I want types and terms to be distinct. There is a decoding operation that injects terms into types, together with the appropriate typing rules.

[†]Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

Russell/Tarski Universes

In my work, I want types and terms to be distinct. There is a decoding operation that injects terms into types, together with the appropriate typing rules.

Gilbert et. al.[†] don't have this decoding operation as their types and terms are merged into expressions.

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

Russell/Tarski Universes

In my work, I want types and terms to be distinct. There is a decoding operation that injects terms into types, together with the appropriate typing rules.

Gilbert et. al.[†] don't have this decoding operation as their types and terms are merged into expressions.

Merging types with terms allows for **Russell** universes, and keeping them separate forces a **Tarski** presentation.

[†] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". 2019.

Problem.

Problem.

Controlling relevance in Russell style:

$$\frac{\ldots \quad \Gamma \vdash A : \textsc{Type} \quad \ldots}{A \text{ is proof-relevant}} \qquad \frac{\ldots \quad \Gamma \vdash A : \textsc{sProp} \quad \ldots}{A \text{ is proof-irrelevant}}$$

Problem.

Controlling relevance in Russell style:

$$\frac{\ldots \quad \Gamma \vdash A : \textsc{Type} \quad \ldots}{A \text{ is proof-relevant}} \qquad \frac{\ldots \quad \Gamma \vdash A : \textsc{sProp} \quad \ldots}{A \text{ is proof-irrelevant}}$$

Problem. Because I don't merge types and terms in expressions, I can't even write $A : \textsc{Prop}$ as above.

Problem.

Controlling relevance in Russell style:

$$\frac{\ldots \quad \Gamma \vdash A : \textsc{Type} \quad \ldots}{A \text{ is proof-relevant}} \qquad \frac{\ldots \quad \Gamma \vdash A : \textsc{sProp} \quad \ldots}{A \text{ is proof-irrelevant}}$$

Problem. Because I don't merge types and terms in expressions, I can't even write $A : \textsc{Prop}$ as above.

All I really want is an equivalent of the following rule

$$\frac{\Gamma \vdash A : \textsc{sProp} \quad \Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x \equiv y : A}$$

Solution.

Solution.

All I really want is an equivalent of the following rule

$$\frac{\Gamma \vdash A : \textsc{sProp} \quad \Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x \equiv y : A}$$

Solution.

All I really want is an equivalent of the following rule

$$\frac{\Gamma \vdash A : \textsc{sProp} \quad \Gamma \vdash x : A \quad \Gamma \vdash y : A}{\Gamma \vdash x \equiv y : A}$$

In this rule, all I really need to know if whether types are relevant or not.

Bake the relevance information into the syntax.

Baking relevance information into the syntax?

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs
- Variables (relevant)

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs
- Variables (relevant)
- Variables (irrelevant)

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs
- Variables (relevant)
- Variables (irrelevant)
- Types (relevant)

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs
- Variables (relevant)
- Variables (irrelevant)
- Types (relevant)
- Types (irrelevant)

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs
- Variables (relevant)
- Variables (irrelevant)
- Types (relevant)
- Types (irrelevant)
- Terms (relevant)

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs
- Variables (relevant)
- Variables (irrelevant)
- Types (relevant)
- Types (irrelevant)
- Terms (relevant)
- Terms (irrelevant)

Baking relevance information into the syntax?

For dependent type theory (without proof irrelevance), in my mechanization, I define contexts, variable, types and terms.

To "bake relevance information into the syntax", the idea is to split types and terms along the relevant/irrelevant distinction. We get

- Contexs
- Variables (relevant)
- Variables (irrelevant)
- Types (relevant)
- Types (irrelevant)
- Terms (relevant)
- Terms (irrelevant)

Sounds naïve...
Next section: how this makes sense from the point of view of low-dimensional category theory!

Idea

Idea

Introducing dimension fixes the issue.

Idea

Introducing dimension fixes the issue.

Insight from Sophie's theory of higher dimensional structures:

Idea

Introducing dimension fixes the issue.

Insight from Sophie's theory of higher dimensional structures:

- Dimension (sizing) is intrinsic.

Idea

Introducing dimension fixes the issue.

Insight from Sophie's theory of higher dimensional structures:

- Dimension (sizing) is intrinsic.
- The formation rule of identity types decreases (size and) dimension.

Idea

Introducing dimension fixes the issue.

Insight from Sophie's theory of higher dimensional structures:

- Dimension (sizing) is intrinsic.
- The formation rule of identity types decreases (size and) dimension.

The formation rule of identity types should look like

$$\frac{\Gamma \vdash A \quad n\text{-type} \qquad \Gamma \vdash x : A \qquad \Gamma \vdash y : A}{\Gamma \vdash x = y \quad (n-1)\text{-type}}$$

Idea

Introducing dimension fixes the issue.

Insight from Sophie's theory of higher dimensional structures:

- Dimension (sizing) is intrinsic.
- The formation rule of identity types decreases (size and) dimension.

The formation rule of identity types should look like

$$\frac{\Gamma \vdash A \quad n\text{-type} \qquad \Gamma \vdash x : A \qquad \Gamma \vdash y : A}{\Gamma \vdash x = y \quad (n-1)\text{-type}}$$

In this work the relevant dimensions are -1 and 0 — propositions and sets.

Dimension

Dimension

For our type theory with definitional proof-irrelevance, we will only need two dimensions: one for sets, and one for propositions.

Dimension

For our type theory with definitional proof-irrelevance, we will only need two dimensions: one for sets, and one for propositions.

Sets are 0-dimensional, therefore the identity types of points of sets are (-1)-dimensional. We denote dimension 0 and -1 as follows

Dimension

For our type theory with definitional proof-irrelevance, we will only need two dimensions: one for sets, and one for propositions.

Sets are 0-dimensional, therefore the identity types of points of sets are (-1)-dimensional. We denote dimension 0 and -1 as follows

$$\boxed{\mathsf{Dim} : \mathsf{Set}}$$

$$\frac{}{\mathsf{-1d} : \mathsf{Dim}} \qquad \frac{}{\mathsf{+0d} : \mathsf{Dim}}$$

Dimension

For our type theory with definitional proof-irrelevance, we will only need two dimensions: one for sets, and one for propositions.

Sets are 0-dimensional, therefore the identity types of points of sets are (-1)-dimensional. We denote dimension 0 and -1 as follows

$$\boxed{\mathsf{Dim} : \mathsf{Set}}$$

$$\frac{}{\mathsf{-1d} : \mathsf{Dim}} \qquad \frac{}{\mathsf{+0d} : \mathsf{Dim}}$$

The well-scoped, untyped syntax now has signature

$$\begin{aligned}
&\mathsf{data}\ \mathsf{Ctx} &&: \mathsf{Set}\\
&\mathsf{data}\ \mathsf{Var} &&: \mathsf{Ctx} \to \mathsf{Dim} \to \mathsf{Set}\\
&\mathsf{data}\ \mathsf{Typ} &&: \mathsf{Ctx} \to \mathsf{Dim} \to \mathsf{Set}\\
&\mathsf{data}\ \mathsf{Trm} &&: \mathsf{Ctx} \to \mathsf{Dim} \to \mathsf{Set}
\end{aligned}$$

Sets and Propositions

Sets and Propositions

P : Typ Γ -1d $\quad\quad$ $\Gamma \vdash P$ $\quad\quad$ P is a proposition

Sets and Propositions

$P : \mathsf{Typ}\ \Gamma\ \mathsf{-1d}$	$\Gamma \vdash P$	P is a proposition
$X : \mathsf{Typ}\ \Gamma\ \mathsf{+0d}$	$\Gamma \vdash X$	X is a set

Sets and Propositions

P : Typ Γ -1d	$\Gamma \vdash P$	P is a proposition
X : Typ Γ +0d	$\Gamma \vdash X$	X is a set
p : Trm Γ -1d	$\Gamma \vdash p :: P$	p is a proof of P

Sets and Propositions

$P : \mathsf{Typ}\ \Gamma\ \text{-}1\mathsf{d}$	$\Gamma \vdash P$	P is a proposition
$X : \mathsf{Typ}\ \Gamma\ +0\mathsf{d}$	$\Gamma \vdash X$	X is a set
$p : \mathsf{Trm}\ \Gamma\ \text{-}1\mathsf{d}$	$\Gamma \vdash p :: P$	p is a proof of P
$x : \mathsf{Trm}\ \Gamma\ +0\mathsf{d}$	$\Gamma \vdash x :: P$	x is a point of X

Sets and Propositions

$P : \mathsf{Typ}\ \Gamma\ \text{-}1\mathsf{d}$	$\Gamma \vdash P$	P is a proposition
$X : \mathsf{Typ}\ \Gamma\ +0\mathsf{d}$	$\Gamma \vdash X$	X is a set
$p : \mathsf{Trm}\ \Gamma\ \text{-}1\mathsf{d}$	$\Gamma \vdash p :: P$	p is a proof of P
$x : \mathsf{Trm}\ \Gamma\ +0\mathsf{d}$	$\Gamma \vdash x :: P$	x is a point of X

The statement of proof-irrelevance for propositions is now

$$\frac{\Gamma : \mathsf{Ctx} \quad P : \mathsf{Typ}\ \Gamma\ \text{-}1\mathsf{d} \quad p : \mathsf{Trm}\ \Gamma\ \text{-}1\mathsf{d} \quad q : \mathsf{Trm}\ \Gamma\ \text{-}1\mathsf{d}}{\Gamma \vdash \quad \Gamma \vdash P \quad \Gamma \vdash p :: P \quad \Gamma \vdash q :: P}$$
$$\Gamma \vdash p \equiv q :: P$$

Sets and Propositions

$P : \mathrm{Typ}\ \Gamma\ \text{-1d}$	$\Gamma \vdash P$	P is a proposition
$X : \mathrm{Typ}\ \Gamma\ +0\mathrm{d}$	$\Gamma \vdash X$	X is a set
$p : \mathrm{Trm}\ \Gamma\ \text{-1d}$	$\Gamma \vdash p :: P$	p is a proof of P
$x : \mathrm{Trm}\ \Gamma\ +0\mathrm{d}$	$\Gamma \vdash x :: P$	x is a point of X

The statement of proof-irrelevance for propositions is now

$$\frac{\Gamma : \mathrm{Ctx} \quad P : \mathrm{Typ}\ \Gamma\ \text{-1d} \quad p : \mathrm{Trm}\ \Gamma\ \text{-1d} \quad q : \mathrm{Trm}\ \Gamma\ \text{-1d}}{\Gamma \vdash \quad \Gamma \vdash P \quad \Gamma \vdash p :: P \quad \Gamma \vdash q :: P} \qquad \frac{}{\Gamma \vdash p \equiv q :: P}$$

And identity types for sets are now

$$\frac{\Gamma : \mathrm{Ctx} \quad X : \mathrm{Typ}\ \Gamma\ +0\mathrm{d} \quad x : \mathrm{Trm}\ \Gamma\ +0\mathrm{d} \quad y : \mathrm{Trm}\ \Gamma\ +0\mathrm{d}}{\Gamma \vdash \quad \Gamma \vdash X \quad \Gamma \vdash x :: X \quad \Gamma \vdash y :: X} \qquad \frac{}{\Gamma \vdash x \simeq y}$$

Dimension-Generic Behavior

Dimension-Generic Behavior

More insight from Sophie's work: The formation rule of Π-types should look like

$$\frac{\Gamma \vdash A \quad n\text{-type} \qquad \Gamma, x : A \vdash B \quad m\text{-type}}{\Gamma \vdash \Pi x : A.\, B \quad m\text{-type}}$$

Dimension-Generic Behavior

More insight from Sophie's work: The formation rule of Π-types should look like

$$\frac{\Gamma \vdash A \quad n\text{-type} \qquad \Gamma, x : A \vdash B \quad m\text{-type}}{\Gamma \vdash \Pi\, x : A.\, B \quad m\text{-type}}$$

The mechanized definition of the above is exactly the same: there is no reference to the specific value of the dimensions.

$$\frac{d_0\ d_1 : \mathsf{Dim} \qquad \begin{array}{c}\Gamma : \mathsf{Ctx} \\ \Gamma \vdash\end{array} \qquad \begin{array}{c}A : \mathsf{Typ}\ \Gamma\ d_0 \\ \Gamma \vdash B\end{array} \qquad \begin{array}{c}B : \mathsf{Typ}\ \Gamma\ ,\ A\ d_1 \\ \Gamma\ ,\ B \vdash B\end{array}}{\Gamma \vdash \Pi\ A\ B}$$

Conclusion

Conclusion

In this presentation, I

Conclusion

In this presentation, I

- Recalled the reference method for the implementation of proof-irrelevance in type theory.

Conclusion

In this presentation, I

- Recalled the reference method for the implementation of proof-irrelevance in type theory.
- Presented my approach to mechanization of type theory, and showed how I couldn't replicate the above method 1:1.

Conclusion

In this presentation, I

- Recalled the reference method for the implementation of proof-irrelevance in type theory.
- Presented my approach to mechanization of type theory, and showed how I couldn't replicate the above method 1:1.
- Gave an apparently naïve solution to recover definitional proof-irrelevance.

Conclusion

In this presentation, I

- Recalled the reference method for the implementation of proof-irrelevance in type theory.
- Presented my approach to mechanization of type theory, and showed how I couldn't replicate the above method 1:1.
- Gave an apparently naïve solution to recover definitional proof-irrelevance.
- Showed how this solution made gemetric sense, following Sophie's insights.

Conclusion

In this presentation, I

- Recalled the reference method for the implementation of proof-irrelevance in type theory.
- Presented my approach to mechanization of type theory, and showed how I couldn't replicate the above method 1:1.
- Gave an apparently naïve solution to recover definitional proof-irrelevance.
- Showed how this solution made gemetric sense, following Sophie's insights.

The sources are available online: https://github.com/thejohncrafter/SSTT26/.

Bibliography

[1] Sophie d'Espalungue d'Arros. "Operads in 2-categories and models of structure interchange". Theses. Université de Lille, Dec. 2023. URL: https://theses.hal.science/tel-04617115.

[2] Gaëtan Gilbert et al. "Definitional proof-irrelevance without K". In: *Proc. ACM Program. Lang.* 3.POPL (2019), 3:1–3:28. DOI: 10.1145/3290316. URL: https://doi.org/10.1145/3290316.

[3] Julien Marquet-Wagner. "Induction-Induction for Intrinsically Well-Scoped Syntaxes". In: *TYPES 2026* (2026). Forthcoming.