Towards a Higher Structure Identity Principle

Benedikt Ahrens

with Paige R. North, Michael Shulman, Dimitris Tsementzis

work in progress

Indiscernability of identicals

Indiscernability of identicals

$$x = y \quad \rightarrow \quad \forall P\,(P(x) \leftrightarrow P(y))$$

- Reasoning **in logic** is invariant under equality
- **In mathematics**, reasoning should be invariant under weaker notion of sameness!

The equivalence principle

Equivalence principle

Reasoning in mathematics should be **invariant under** the appropriate notion of **sameness**.

Notion of sameness depends on the objects under consideration:

- **equal** numbers, functions,...
- **isomorphic** sets, groups, rings,...
- **equivalent** categories
- **biequivalent** bicategories
- ...

Violating the equivalence principle

We can easily **violate** this principle:

Exercise

Find a statement about categories that is not invariant under the equivalence of categories

Violating the equivalence principle

We can easily **violate** this principle:

Exercise

Find a statement about categories that is not invariant under the equivalence of categories

A solution

"The category $\mathscr{C}$ has exactly one object."

Maybe this statement is simply silly!

A language for invariant properties

Michael Makkai, *Towards a Categorical Foundation of Mathematics*:

> *The basic character of the Principle of Isomorphism is that of a* ***constraint on the language*** *of Abstract Mathematics; a welcome one, since it provides for the separation of sense from nonsense.*

A language for invariant properties

Michael Makkai, *Towards a Categorical Foundation of Mathematics*:

> *The basic character of the Principle of Isomorphism is that of a* ***constraint on the language*** *of Abstract Mathematics; a welcome one, since it provides for the separation of sense from nonsense.*

Makkai's goal

to devise a language in which only invariant properties can be expressed

$\rightsquigarrow$ First Order Logic with Dependent Sorts (FOLDS)

FOLDS is quite similar to dependent type theory

Univalent foundations and the equivalence principle

Goal

- Univalent foundations as an “invariant language”
- Invariance not only for statements, but also for constructions

In this talk

- Introduction to univalent foundations
- Invariance for set-level structures (Structure Identity Principle)
- Invariance for **univalent** categories
- Our goal: generalizing “univalence” condition to other (higher-categorical) structures and prove invariance for univalent such structures

Outline

1. Dependent type theory
2. Some important concepts in univalent foundations
3. The Structure Identity Principle for set-level structures
4. Category theory in univalent foundations
5. FOLDS categories
6. FOLDS structures

Outline

1. Dependent type theory
2. Some important concepts in univalent foundations
3. The Structure Identity Principle for set-level structures
4. Category theory in univalent foundations
5. FOLDS categories
6. FOLDS structures

Overview of type theory

(Dependent) Type theory

- Is a (functional programming) language of types and terms
- Has infrastructure to write mathematical statements and proofs

Write $a : A$ to say that term a has type A (e.g., $1 : \mathsf{Nat}$)

Writing well-typed programs

In type theory, both the activities of

- implementing an algorithm
- proving a mathematical statement

are done by writing well-typed programs.

Dependent types and functions

Examples of dependent types

- $\mathsf{Vect}(A, n)$ — type of vectors of length $n : \mathsf{Nat}$ of elements in type A
- $\mathsf{GrpStr}(X)$ — type of group structures on a set X

Examples of dependent functions

$$\mathsf{zero} : \prod_{m:\mathsf{Nat}} \mathsf{Vect}(\mathsf{Nat}, m)$$

$$\mathsf{tail} : \prod_{n:\mathsf{Nat}} \mathsf{Vect}(A, 1+n) \to \mathsf{Vect}(A, n)$$

$$\mathsf{GrpStrTransfer} : \prod_{X,Y:\mathsf{Set}} (X \cong Y) \times \mathsf{GrpStr}(X) \to \mathsf{GrpStr}(Y)$$

Overview of types in type theory

Type former	Notation	(special case)	canonical term
Inhabitant	$a : A$		
Dependent type	$x : A \vdash B(x)$		
Sigma type	$\sum_{x:A} B(x)$	$A \times B$	(a, b)
Product type	$\prod_{x:A} B(x)$	$A \to B$	$\lambda(x : A).b$
Coproduct type	$A + B$		$\mathsf{inl}(a)$, $\mathsf{inr}(b)$
Identity type	$a \rightsquigarrow_A b$		$\mathsf{refl}(a) : a \rightsquigarrow a$
Universe	Type		
Base types	Nat, Bool, 1, 0		

Some terms that can be defined

Induction principle for $a \rightsquigarrow b$

To define a function

$$f : \prod_{(x,y:A)} \prod_{(p:x \rightsquigarrow y)} C(x,y,p)$$

it suffices to specify its image on $(x, x, \mathsf{refl}(x))$.

- $\mathsf{false} \rightsquigarrow_{\mathsf{Bool}} \mathsf{false}$
- $\mathsf{sym} : \prod_{x,y:A} (x \rightsquigarrow y) \to (y \rightsquigarrow x)$
- $\mathsf{trans} : \prod_{x,y,z:A} (x \rightsquigarrow y) \times (y \rightsquigarrow z) \to (x \rightsquigarrow z)$

Transport

$$\mathsf{transport}^A : \prod_{x,y:A} (x \rightsquigarrow y) \to \prod_{B:A \to \mathsf{Type}} \big(B(x) \leftrightarrow B(y)\big)$$

Identities vs equalities

Inhabitants of $a \rightsquigarrow a'$ behave like equality in many ways

- reflexivity, symmetry, transitivity
- transport

Inhabitants of $a \rightsquigarrow a'$ behave **un**like equality

- Can iterate identity type
- Cannot show that any two identities are identical

Outline

1. Dependent type theory
2. Some important concepts in univalent foundations
3. The Structure Identity Principle for set-level structures
4. Category theory in univalent foundations
5. FOLDS categories
6. FOLDS structures

The important features of univalent foundations

Homotopy levels

- Stratification of types according to “complexity” of their identity types
- Logic: notion of propositions given by one layer of this hierarchy

Univalence axiom

Specifies the previously unspecified identity type of a universe

Logic in univalent foundations

Propositions

Type A is a proposition when we can define a term of type

$$\mathsf{isProp}(A) :\equiv \prod_{x,y:A} x \rightsquigarrow y$$

$$\mathsf{Prop} :\equiv \sum_{X:\mathsf{Type}} \mathsf{isProp}(X) \qquad \text{world of propositions}$$

- implication $A \to B$
- conjunction $A \times B$
- predicate $B : A \to \mathsf{Prop}$
- universal quantification $\prod_{a:A} B(a)$
- disjunction and existential need extra care

Contractible types, propositions and sets

- A is **contractible**

$$\mathsf{isContr}(A) :\equiv \sum_{x:A} \prod_{y:A} y \rightsquigarrow x$$

- A is a **proposition**

$$\mathsf{isProp}(A) :\equiv \prod_{x,y:A} x \rightsquigarrow y$$

- A is a **set**

$$\mathsf{isSet}(A) :\equiv \prod_{x,y:A} \mathsf{isProp}(x \rightsquigarrow y)$$

$$\mathsf{Prop} :\equiv \sum_{X:\mathsf{Type}} \mathsf{isProp}(X) \qquad \mathsf{Set} :\equiv \sum_{X:\mathsf{Type}} \mathsf{isSet}(X)$$

Contractible types, propositions and sets

- A is **contractible**

$$\mathsf{isContr}(A) \;:\equiv\; \sum_{x:A} \prod_{y:A} y \rightsquigarrow x$$

- A is a **proposition**

$$\mathsf{isProp}(A) \;:\equiv\; \prod_{x,y:A} \mathsf{isContr}(x \rightsquigarrow y)$$

- A is a **set**

$$\mathsf{isSet}(A) \;:\equiv\; \prod_{x,y:A} \mathsf{isProp}(x \rightsquigarrow y)$$

$$\mathsf{Prop} \;:\equiv\; \sum_{X:\mathsf{Type}} \mathsf{isProp}(X) \qquad \mathsf{Set} \;:\equiv\; \sum_{X:\mathsf{Type}} \mathsf{isSet}(X)$$

Equivalences

Definition

A map $f : A \to B$ is an **equivalence** if it has contractible fibers, i.e.,

$$\mathsf{isequiv}(f) \quad :\equiv \quad \prod_{b:B} \mathsf{isContr}\left(\sum_{a:A} f(a) \rightsquigarrow b\right)$$

The type of equivalences:

$$A \simeq B \quad :\equiv \quad \sum_{f:A\to B} \mathsf{isequiv}(f)$$

Transport revisited

$$\mathsf{transport}^A : \prod_{x,y:A} (x \rightsquigarrow y) \to \prod_{B:A\to \mathsf{Type}} \big(B(x) \simeq B(y)\big)$$

The path type of pairs

Can construct equivalences

- for $s, t : A \times B$

$$\left(s \rightsquigarrow t\right) \simeq \left(\left(\mathsf{fst}(s) \rightsquigarrow \mathsf{fst}(t)\right) \times \left(\mathsf{snd}(s) \rightsquigarrow \mathsf{snd}(t)\right)\right)$$

- for $s, t : \sum_{x:A} B(x)$

$$\left(s \rightsquigarrow t\right) \simeq \left(\sum_{e:\mathsf{fst}(s) \rightsquigarrow \mathsf{fst}(t)} \mathsf{transport}^{B}\left(e, \mathsf{snd}(s)\right) \rightsquigarrow \mathsf{snd}(t)\right)$$

The path type of pairs

Can construct equivalences

- for $s, t : A \times B$

$$\left(s \rightsquigarrow t\right) \simeq \left(\left(\mathsf{fst}(s) \rightsquigarrow \mathsf{fst}(t)\right) \times \left(\mathsf{snd}(s) \rightsquigarrow \mathsf{snd}(t)\right)\right)$$

- for $s, t : \sum_{x:A} B(x)$

$$\left(s \rightsquigarrow t\right) \simeq \left(\sum_{e:\mathsf{fst}(s) \rightsquigarrow \mathsf{fst}(t)} \mathsf{transport}^{B}\left(e, \mathsf{snd}(s)\right) \rightsquigarrow \mathsf{snd}(t)\right)$$

Can other path types be characterized similarly?

Axioms to characterize some path types

Axiom of function extensionality for $f, g : A \to B$

$$\left(f \rightsquigarrow g\right) \simeq \left(\prod_{a:A} f(a) \rightsquigarrow g(a)\right)$$

Univalence axiom for $A, B : \mathsf{Type}$

$$\left(A \rightsquigarrow B\right) \simeq \left(A \simeq B\right)$$

More precisely: canonical map $\to$ is an equivalence.

Theorem (Voevodsky)

Univalence axiom implies function extensionality

Outline

1. Dependent type theory
2. Some important concepts in univalent foundations
3. The Structure Identity Principle for set-level structures
4. Category theory in univalent foundations
5. FOLDS categories
6. FOLDS structures

Transport along isomorphism

Have:

$$\mathsf{transport}_{x,y} : (x \rightsquigarrow y) \to \prod_{B:A\to\mathsf{Type}} \big(B(x) \simeq B(y)\big)$$

Want:

$$\mathsf{transport}_{x,y} : (x \cong y) \to \prod_{B:A\to\mathsf{Type}} \big(B(x) \simeq B(y)\big)$$

Suffices:

$$(x \rightsquigarrow y) \;\simeq\; (x \cong y)$$

Transport along isomorphism

Have:

$$\mathsf{transport}_{x,y} : (x \rightsquigarrow y) \to \prod_{B:A\to\mathsf{Type}} \big(B(x) \simeq B(y)\big)$$

Want:

$$\mathsf{transport}_{x,y} : (x \cong y) \to \prod_{B:A\to\mathsf{Type}} \big(B(x) \simeq B(y)\big)$$

Suffices:

$$(x \rightsquigarrow y) \xrightarrow{\simeq} (x \cong y)$$

Monoids in type theory

In type theory, a monoid is a tuple $(M, \mu, e, \alpha, \lambda, \rho)$ where

1. $M : \mathsf{Set}$
2. $\mu : M \times M \to M$
3. $e : M$
4. $\alpha : \Pi_{(a,b,c:M)} \mu(\mu(a,b),c) \rightsquigarrow \mu(a,\mu(b,c))$
5. $\lambda : \Pi_{(a:M)} \mu(e,a) \rightsquigarrow a$
6. $\rho : \Pi_{(a:M)} \mu(a,e) \rightsquigarrow a$

Monoids in type theory

In type theory, a monoid is a tuple $(M, \mu, e, \alpha, \lambda, \rho)$ where

1. $M : \mathsf{Set}$
2. $\mu : M \times M \to M$
3. $e : M$
4. $\alpha : \Pi_{(a,b,c:M)} \mu(\mu(a,b),c) \rightsquigarrow \mu(a,\mu(b,c))$
5. $\lambda : \Pi_{(a:M)} \mu(e,a) \rightsquigarrow a$
6. $\rho : \Pi_{(a:M)} \mu(a,e) \rightsquigarrow a$

Why $M : \mathsf{Set}$?

Monoids in type theory

In type theory, a monoid is a tuple $(M, \mu, e, \alpha, \lambda, \rho)$ where

1. $M : \mathsf{Set}$
2. $\mu : M \times M \to M$
3. $e : M$
4. $\alpha : \Pi_{(a,b,c:M)} \mu(\mu(a,b),c) \rightsquigarrow \mu(a,\mu(b,c))$
5. $\lambda : \Pi_{(a:M)} \mu(e,a) \rightsquigarrow a$
6. $\rho : \Pi_{(a:M)} \mu(a,e) \rightsquigarrow a$

Why $M : \mathsf{Set}$?

Abstractly, a monoid is a (dependent) pair $(data, proof)$ where

- $data$ is 1.–3.
- $proof$ is 4.–6.

The type of monoids

- We want two monoids $(data, proof)$ and $(data', proof')$ to be the same if $data$ is the same as $data'$.
- This is guaranteed when the types of $proof$ and $proof'$ are **propositions**.
- This in turn guaranteed when the underlying type M is a **set**.

Summarily:

$$\mathsf{Monoid} :\equiv \sum_{(M:\mathsf{Set})} \sum_{(\mu,e):\mathsf{MonoidStr}(M)} \mathsf{MonoidAxioms}(M,(\mu,e))$$

Can show

$$\mathsf{isProp}(\mathsf{MonoidAxioms}(M,(\mu,e)))$$

Monoid isomorphisms

Given $\mathbf{M} \equiv (M, \mu, e, \alpha, \lambda, \rho)$ and $\mathbf{M}' \equiv (M', \mu', e', \alpha', \lambda', \rho')$, a **monoid isomorphism** is a bijection $f : M \cong M'$ preserving μ and e.

Monoid isomorphisms

Given $\mathbf{M} \equiv (M, \mu, e, \alpha, \lambda, \rho)$ and $\mathbf{M}' \equiv (M', \mu', e', \alpha', \lambda', \rho')$, a **monoid isomorphism** is a bijection $f : M \cong M'$ preserving μ and e.

$$
\begin{aligned}
\mathbf{M} \rightsquigarrow \mathbf{M}' &\simeq (M, \mu, e) \rightsquigarrow (M', \mu', e') \\
&\simeq \sum_{p : M \rightsquigarrow M'} (\mathsf{transport}^{Y \mapsto (Y \times Y \to Y)}(p, \mu) \rightsquigarrow \mu') \\
&\qquad \times (\mathsf{transport}^{Y \mapsto Y}(p, e) \rightsquigarrow e') \\
&\simeq \sum_{f : M \cong M'} \left(f \circ m \circ (f^{-1} \times f^{-1}) \rightsquigarrow m'\right) \\
&\qquad \times (f \circ e \rightsquigarrow e') \\
&\simeq \mathbf{M} \cong \mathbf{M}'
\end{aligned}
$$

Transport along monoid isomorphism

We now have two ingredients:

1. $$\mathsf{transport}_{\mathbf{M},\mathbf{M}'} : (\mathbf{M} \rightsquigarrow \mathbf{M}') \to \prod_{B:\mathsf{Monoid}\to\mathsf{Type}} \left(B(\mathbf{M}) \simeq B(\mathbf{M}')\right)$$

2. $$(\mathbf{M} \rightsquigarrow \mathbf{M}') \;\simeq\; (\mathbf{M} \cong \mathbf{M}')$$

Composing these, we get

$$\mathsf{transport}_{\mathbf{M},\mathbf{M}'} : (\mathbf{M} \cong \mathbf{M}') \to \prod_{B:\mathsf{Monoid}\to\mathsf{Type}} \left(B(\mathbf{M}) \simeq B(\mathbf{M}')\right)$$

I.e., any structure on monoids that can be expressed in univalent type theory can be transported along isomorphism of monoids.

Structure Identity Principle

Structure Identity Principle (Aczel, Coquand&Danielsson)

For many set-level structures in univalent foundations, paths are isomorphisms.

Examples include:

- monoids, groups, rings
- posets
- discrete fields
- sets with fixpoint operator

Structure Identity Principle

Structure Identity Principle (Aczel, Coquand&Danielsson)

For many set-level structures in univalent foundations, paths are isomorphisms.

Examples include:

- monoids, groups, rings
- posets
- discrete fields
- sets with fixpoint operator

What about **categories**?

Outline

1. Dependent type theory
2. Some important concepts in univalent foundations
3. The Structure Identity Principle for set-level structures
4. Category theory in univalent foundations
5. FOLDS categories
6. FOLDS structures

EP for categories

Conjecture

For categories $\mathscr{C}$ and $\mathscr{D}$, the canonical map

$$(\mathscr{C} \rightsquigarrow \mathscr{D}) \;\to\; \mathsf{AdjEquiv}(\mathscr{C}, \mathscr{D})$$

is an equivalence.

EP for categories

Conjecture

For categories $\mathscr{C}$ and $\mathscr{D}$, the canonical map

$$(\mathscr{C} \rightsquigarrow \mathscr{D}) \;\to\; \mathsf{AdjEquiv}(\mathscr{C}, \mathscr{D})$$

is an equivalence.

Counter-example

Categories in univalent foundations

Definition

A **category** $\mathscr{C}$ is given by

- a **type** $\mathscr{C}_0$: Type of **objects**
- for any $a, b : \mathscr{C}_0$, a **set** $\mathscr{C}(a,b)$: Set of **morphisms**
- operations: identity & composition

$$1_a : \mathscr{C}(a,a)$$
$$(\circ)_{a,b,c} : \mathscr{C}(b,c) \times \mathscr{C}(a,b) \to \mathscr{C}(a,c)$$

- axioms: unitality & associativity

$$1 \circ f \rightsquigarrow f \qquad f \circ 1 \rightsquigarrow f \qquad (h \circ g) \circ f \rightsquigarrow h \circ (g \circ f)$$

From paths to isomorphisms

Definition (univalent category)

For a category $\mathscr{C}$ we define

$$\mathsf{idtoiso} : \prod_{a,b:\mathscr{C}_0} (a \rightsquigarrow b) \to \mathsf{iso}(a,b)$$

$$\mathsf{idtoiso}(a,a,\mathsf{refl}(a)) \quad :\equiv \quad 1_a$$

We call the category $\mathscr{C}$ **univalent** if, for any objects $a, b : \mathscr{C}_0$,

$$\mathsf{idtoiso}_{a,b} : (a \rightsquigarrow b) \to \mathsf{iso}(a,b)$$

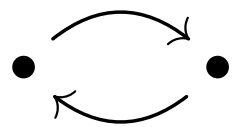
is an equivalence of types.

Examples of univalent categories

- Set
- Groups, rings, . . . (Structure Identity Principle)
- Functor category $[\mathscr{C}, \mathscr{D}]$, if $\mathscr{D}$ is univalent
- Full subcategories of univalent categories
- A preorder is univalent iff it is antisymmetric
- If X is of h-level 3, then there is a univalent category with X as objects and $\hom(x,y) \;:\equiv\; (x \rightsquigarrow y)$
- If $\mathscr{C}$ is univalent, then the category of cones of shape $F : \mathscr{J} \to \mathscr{C}$ is
  - $\rightsquigarrow$ limits (limiting cones) in a univalent category are unique **up to paths**

Non-univalent categories

- Any "chaotic" category $\mathscr{C}$ with $\mathscr{C}(x,y) \simeq 1$, for $\mathscr{C}_0$ not a proposition



- Any chaotic category $\mathscr{C}$ with an object $c : \mathscr{C}_0$ is **equivalent** to the terminal category $\mathbf{1}$
  - $\rightsquigarrow$ a category can be equivalent to a univalent one without being univalent itself

(Adjoint) equivalence of categories

An **equivalence** $\mathscr{C} \simeq \mathscr{D}$ is given by

- a functor $F : \mathscr{C} \to \mathscr{D}$
- a functor $G : \mathscr{D} \to \mathscr{C}$
- a natural isomorphism $\eta : 1_{\mathscr{C}} \xrightarrow{\cong} GF$
- a natural isomorphism $\epsilon : FG \xrightarrow{\cong} 1_{\mathscr{D}}$

(F, G, η, ϵ) is an adjoint equivalence if F and G form an adjunction.

(Adjoint) equivalence of categories

An **equivalence** $\mathscr{C} \simeq \mathscr{D}$ is given by

- a functor $F : \mathscr{C} \to \mathscr{D}$
- a functor $G : \mathscr{D} \to \mathscr{C}$
- a natural isomorphism $\eta : 1_{\mathscr{C}} \xrightarrow{\cong} GF$
- a natural isomorphism $\epsilon : FG \xrightarrow{\cong} 1_{\mathscr{D}}$

(F, G, η, ϵ) is an adjoint equivalence if F and G form an adjunction.

Theorem

*For **univalent** categories $\mathscr{C}$ and $\mathscr{D}$, the canonical map*

$$(\mathscr{C} \rightsquigarrow \mathscr{D}) \;\to\; \mathsf{AdjEquiv}(\mathscr{C}, \mathscr{D})$$

is an equivalence.

Goal

Prove a similar theorem for other structures.

Envisioned result

Given a signature $\mathscr{L}$, and two $\mathscr{L}$-univalent $\mathscr{L}$-structures M and N, then

$$(M \rightsquigarrow N) \xrightarrow{\simeq} (M \simeq_{\mathscr{L}} N)$$

Need notions of

- signature
- $\mathscr{L}$-structure for a signature $\mathscr{L}$
- $\mathscr{L}$-equivalence of $\mathscr{L}$-structures
- $\mathscr{L}$-isomorphism and $\mathscr{L}$-univalence

Goal

Prove a similar theorem for other structures.

Envisioned result

Given a signature $\mathscr{L}$, and two $\mathscr{L}$-univalent $\mathscr{L}$-structures M and N, then

$$(M \rightsquigarrow N) \xrightarrow{\simeq} (M \simeq_{\mathscr{L}} N)$$

Need notions of

- signature
- $\mathscr{L}$-structure for a signature $\mathscr{L}$
- $\mathscr{L}$-equivalence of $\mathscr{L}$-structures
- $\mathscr{L}$-isomorphism and $\mathscr{L}$-univalence

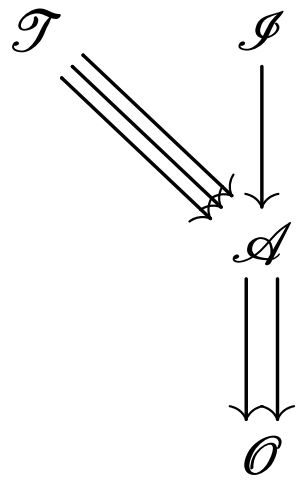
Before, reformulate the example of categories.

Outline

1. Dependent type theory
2. Some important concepts in univalent foundations
3. The Structure Identity Principle for set-level structures
4. Category theory in univalent foundations
5. FOLDS categories
6. FOLDS structures

Makkai's FOLDS

- Signature := inverse category

$\mathscr{L}_{\text{cat}} :=$ 

+ equalities of arrows

- Structures for a signature $\mathscr{L} := \mathscr{L} \to \mathsf{Set}$
- Signature only specifies the data, e.g., of categories
- FOLDS language for axioms

FOLDS structures in type theory

In type theory natural to define a structure for $\mathscr{L}_{\text{cat}}$ to be

- O : Type;
- $A : O \times O \to \text{Type}$;
- $I : \prod_{x:O} A(x,x) \to \text{Type}$; and
- $T : \prod_{x,y,z:O} A(x,y) \to A(y,z) \to A(x,z) \to \text{Type}$.

Here I, T Type-valued predicates "being an identity", "being a composite"

Attention

Axioms not part of $\mathscr{L}_{\text{cat}}$-structures, have to be imposed additionally:

1. Existence and uniqueness of composition
2. Existence of identity arrows
3. Categorical axioms

Comparison 1-univalent FOLDS-cats vs categories

Definition (1-univalent FOLDS-cat)

An $\mathscr{L}_{\text{cat}}$-structure such that

- $I_x(f)$, $T_{x,y,z}(f,g,h)$, and $E_{x,y}(f,g)$ are propositions
- $A(x,y)$ is a set,
- and the axioms 1–3 are satisfied.

Equivalence of types

$$\text{1-univ. FOLDS-cats} \quad \simeq \quad \text{categories}$$

Goal

Define type of isomorphisms $a \cong_{\mathscr{L}_{\text{cat}}} b$ for 1-univ. FOLDS-cats **without referring to the axioms** 1–3

FOLDS isomorphism for categories

Goal

Define type of isomorphisms $a \cong_{\mathscr{L}_{\text{cat}}} b$ for 1-univ. FOLDS-cats **without referring to the axioms** 1–3

Using Yoneda:

- For each $x : O$, an isomorphism $\phi_{x\bullet} : A(x,a) \cong A(x,b)$.
- For each $x,y : O$, $f : A(x,y)$, $g : A(y,a)$, and $h : A(x,a)$, we have

$$T_{x,y,a}(f,g,h) \Leftrightarrow T_{x,y,b}(f, \phi_{y\bullet}(g), \phi_{x\bullet}(h))$$

(means $\phi_{y\bullet}(g) \circ f = \phi_{x\bullet}(g \circ f)$)

Problem: not symmetric, not algorithmic

FOLDS isomorphism for categories

Equivalences between hom-types with a and b substituted into *all* possible "collections of holes":

1. For each $x : O$, an equivalence $\phi_{x\bullet} : A(x,a) \simeq A(x,b)$.
2. For each $z : O$, an equivalence $\phi_{\bullet z} : A(a,z) \simeq A(b,z)$.
3. An isomorphism $\phi_{\bullet\bullet} : A(a,a) \simeq A(b,b)$.

and equivalences between all "relations with holes":

$$
\begin{aligned}
T_{x,y,a}(f,g,h) &\simeq T_{x,y,b}(f, \phi_{y\bullet}(g), \phi_{x\bullet}(h)) \\
T_{x,a,z}(f,g,h) &\simeq T_{x,b,z}(\phi_{x\bullet}(f), \phi_{\bullet z}(g), h) \\
T_{a,z,w}(f,g,h) &\simeq T_{b,z,w}(\phi_{\bullet z}(f), g, \phi_{\bullet w}(h)) \\
T_{x,a,a}(f,g,h) &\simeq T_{x,b,b}(\phi_{x\bullet}(f), \phi_{\bullet\bullet}(g), \phi_{x\bullet}(h)) \\
T_{a,x,a}(f,g,h) &\simeq T_{b,x,b}(\phi_{\bullet x}(f), \phi_{x\bullet}(g), \phi_{\bullet\bullet}(h)) \\
T_{a,a,x}(f,g,h) &\simeq T_{b,b,x}(\phi_{\bullet\bullet}(f), \phi_{\bullet x}(g), \phi_{\bullet x}(h)) \\
T_{a,a,a}(f,g,h) &\simeq T_{b,b,b}(\phi_{\bullet\bullet}(f), \phi_{\bullet\bullet}(g), \phi_{\bullet\bullet}(h)) \\
I_{a,a}(f) &\simeq I_{b,b}(\phi_{\bullet\bullet}(f))
\end{aligned}
$$

Univalent FOLDS categories

Theorem

In any 1-univ. FOLDS cat, the type of FOLDS-isomorphisms $a \cong_{\mathscr{L}_{cat}} b$ *is equivalent to the type of ordinary isomorphisms* $a \cong b$.

Definition

A 1-univ. FOLDS cat **is o-univalent** if for all $a, b : O$,

$$(a \rightsquigarrow b) \;\to\; (a \cong_{\mathscr{L}_{\text{cat}}} b) \text{ FOLDS-isomorphisms}$$

is an equivalence.

Theorem

A 1-univ. FOLDS cat is o-univ. if and only if its corresponding category is a univalent category.

Reminder: 1-univalent FOLDS-categories

A 1-univalent FOLDS-category consists of

- $O : \mathsf{Type}$;
- $A : O \times O \to \mathsf{Type}$;
- $I : \prod_{x:O} A(x,x) \to \mathsf{Type}$;
- $T : \prod_{x,y,z:O} A(x,y) \to A(y,z) \to A(x,z) \to \mathsf{Type}$
- $I_x(f)$ and $T_{x,y,z}(f,g,h)$ are propositions
- $A(x,y)$ is a set
- axioms 1–3

Reminder: 1-univalent FOLDS-categories

A 1-univalent FOLDS-category consists of

- O : Type;
- $A : O \times O \to$ Type;
- $I : \prod_{x:O} A(x,x) \to$ Type;
- $T : \prod_{x,y,z:O} A(x,y) \to A(y,z) \to A(x,z) \to$ Type
- $I_x(f)$ and $T_{x,y,z}(f,g,h)$ are propositions
- $A(x,y)$ is a set
- axioms 1–3

Reminder: 1-univalent FOLDS-categories

A 1-univalent FOLDS-category consists of

- $O : \mathsf{Type}$;
- $A : O \times O \to \mathsf{Type}$;
- $I : \prod_{x:O} A(x,x) \to \mathsf{Type}$;
- $T : \prod_{x,y,z:O} A(x,y) \to A(y,z) \to A(x,z) \to \mathsf{Type}$
- $I_x(f)$ and $T_{x,y,z}(f,g,h)$ are propositions
- $A(x,y)$ is a set
- axioms 1–3

Are implied by

- being "univalent at I and T"

Reminder: 1-univalent FOLDS-categories

A 1-univalent FOLDS-category consists of

- $O : \mathsf{Type}$;
- $A : O \times O \to \mathsf{Type}$;
- $I : \prod_{x:O} A(x,x) \to \mathsf{Type}$;
- $T : \prod_{x,y,z:O} A(x,y) \to A(y,z) \to A(x,z) \to \mathsf{Type}$
- $I_x(f)$ and $T_{x,y,z}(f,g,h)$ are propositions
- $A(x,y)$ is a set
- axioms 1–3

Are implied by

- being "univalent at I and T"
- being "univalent at A"

Univalence at level A for 1-saturated FOLDS cats

Given $f, g : A(a,b)$ in a 1-saturated FOLDS cat, an isomorphism $i : f \cong_{\mathscr{L}_{\text{cat}}} g$ is given by

$$T_{x,a,b}(u,f,v) \simeq T_{x,a,b}(u,g,v)$$
$$T_{a,x,b}(u,v,f) \simeq T_{a,x,b}(u,v,g)$$
$$T_{a,b,x}(f,u,v) \simeq T_{x,a,b}(g,u,v)$$
$$\vdots$$

Observation

These types of equivalences are all propositions, and thus $(f \cong_{\mathscr{L}_{\text{cat}}} g)$ is a proposition.

Theorem

If $(f \rightsquigarrow g) \to (f \cong g)$ is an equivalence for all f, g, then $A(x,y)$ is a set for any x, y.

FOLDS categories

Definition

A **FOLDS-category** consists of the same data and axioms as a 1-saturated, but without T, I, and E being propositions and A being sets.

Have

$$(t \cong_{\mathscr{L}_{\text{cat}}} t') \simeq 1$$

since nothing "above T".

Theorem

In a FOLDS-category, T and E are propositions iff the canonical maps

$$(t = t') \to (t \cong t')$$
$$(i = i') \to (i \cong i')$$

are equivalences for all inhabitants of T and I respectively.

Conclusions from this example

- Notion of FOLDS category only requires $\mathscr{L}_{\text{cat}}$ and axioms 1–3
- Condition on homotopy levels is consequence of univalence at A, T, and I
- Notion of isomorphism and saturation conditions are independent of axioms of a FOLDS category
- (Notion of equivalence of structures does not depend on axioms)

Outline

1. Dependent type theory
2. Some important concepts in univalent foundations
3. The Structure Identity Principle for set-level structures
4. Category theory in univalent foundations
5. FOLDS categories
6. FOLDS structures

Idea: define signatures by recursion on height

- Example of categories emphasizes the significance of 'level-wise' stratification of FOLDS signatures
- Suggests defining recursively the type of signatures of height n
- Defined mutually with many other things

Signatures

- There is exactly one signature of height 0
- A signature $\mathscr{L}$ of height $n+1$ is given by
  - A signature $\mathscr{L}_{\leq n}$ of height n of 'sorts of level up to n'
  - A type $\mathscr{L}_{n+1}$ of 'sorts on level $n+1$'
  - A family of diagrams,

$$\mathscr{L} \to \mathsf{D}(\mathscr{L}_{\leq n})$$

Intuitively, $\mathsf{D}(\mathscr{L}_{\leq n})$ is "well-behaved functors $\mathscr{L}_{\leq n} \to \mathsf{Type}$".

Signatures and structures recursively

- Diagrams

$$\mathsf{D}_0(\mathscr{L}) := 1$$

$$\mathsf{D}_{n+1}(\mathscr{L}) := \sum_{M:\mathscr{L}_0 \to \mathsf{Type}} \mathsf{D}_n(\mathscr{L}'_M)$$

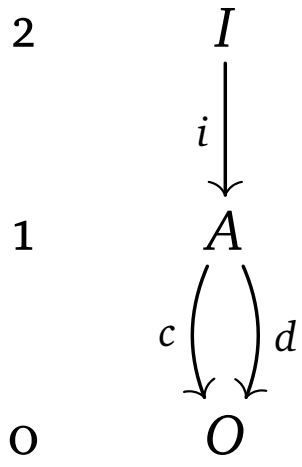
- Derivation:

$$\prod_{n:\mathsf{Nat}} \prod_{\mathscr{L}:\mathsf{Sig}(n+1)} (\mathscr{L}_0 \to \mathsf{Type}) \to \mathsf{Sig}(n)$$

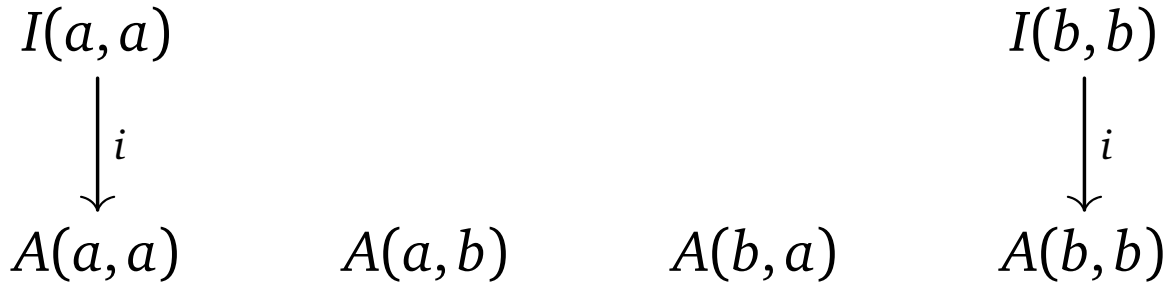
$$(\mathscr{L}, M) \mapsto \mathscr{L}'_M$$

Derivation by example

$$\prod_{n:\mathsf{Nat}} \prod_{\mathscr{L}:\mathsf{Sig}(n+1)} (\mathscr{L}_0 \to \mathsf{Type}) \to \mathsf{Sig}(n)$$



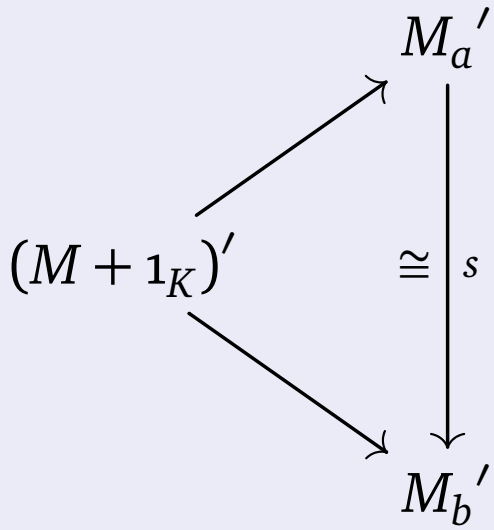
$(\mathscr{L}_{\mathrm{rg}})_0 \equiv \{O\}$
Let M_0 be a (a function picking out) the two-element set $\{a, b\}$. Then $(\mathscr{L})'_{M_0}$ is the following signature, with four sorts of rank 0 and two sorts of rank 1:



$\mathscr{L}$-isomorphism

(Pseudo-)definition

Let M be an $\mathscr{L}$-structure, $K : \mathscr{L}_0$, and $a, b : MK$. The type $a \cong_M b$ of $\mathscr{L}$-isomorphisms from a to b is defined to be the type of levelwise equivalences $s : {M_a}' \cong {M_b}'$ in $\mathsf{D}(\mathscr{L}'_{M_0+1_K})$ under $(M + 1_K)'$, i.e. those s such that the following triangle commutes:



Envisioned results

Envisioned result I

Let $\mathscr{L}$ be of height n and $K : \mathscr{L}_0$. Let M be a univalent $\mathscr{L}$-structure. Then M_0K is of h-level n.

Envisioned result II

The type of univalent structures of $\mathscr{L}$ is of h-level $h(\mathscr{L}) + 1$.

Envisioned result III

Consider $\mathscr{L}$-structures M, N for some signature $\mathscr{L}$ such that M is univalent. Then

$$(M \rightsquigarrow N) \quad \simeq \quad \text{very surjective spans between } M \text{ and } N$$

Conclusions

- In univalent foundations, SIP can be proved for various set-level mathematical structures
- SIP only holds for *univalent* categories
- Pseudo-conjecture inspired by the case of categories: SIP holds for higher-categorical "univalent" structures

Conclusions

- In univalent foundations, SIP can be proved for various set-level mathematical structures
- SIP only holds for *univalent* categories
- Pseudo-conjecture inspired by the case of categories: SIP holds for higher-categorical "univalent" structures

Thanks for your attention!