Categories as Semicategories with Identities

Joshua Chen Tom de Jong Nicolai Kraus Stiéphen Pradal

Functional Programming Lab
University of Nottingham

TYPES 2023, 12-15 June

Motivation

Goal?

Develop ∞-category theory internally to HoTT.

Motivation

Goal?

Develop ∞-category theory internally to HoTT.

Why?

- Describe the natural higher structure of the Universes internally.
- Develop a syntactic theory of higher categories.
- Related to the problem of HoTT eating itself.
- $\cdots$

Motivation

Goal?

Develop ∞-category theory internally to HoTT.

Why?

- Describe the natural higher structure of the Universes internally.
- Develop a syntactic theory of higher categories.
- Related to the problem of HoTT eating itself.
- $\cdots$

Very difficult problem due to coherence issues.

Motivation

How?

1. Define the composition structure $\rightsquigarrow$ ∞-semicategory.
2. Define identities.

Motivation

How?

1. Define the composition structure $\rightsquigarrow$ ∞-semicategory.
2. Define identities.

The type of identity-structure should be a proposition:

- An ∞-semicategory should be an ∞-category in only one way.
- Define ∞-categories as a sub-type of ∞-semicategories.

Motivation

How?

1. Define the composition structure $\rightsquigarrow$ ∞-semicategory.
2. Define identities.

The type of identity-structure should be a proposition:

- An ∞-semicategory should be an ∞-category in only one way.
- Define ∞-categories as a sub-type of ∞-semicategories.

Generalisation:

In this talk we will focus on wild categories:

- No higher coherence conditions.
- No truncation conditions.

$\rightsquigarrow$ Wild categories generalise ∞-categories.

Outline

1. Wild semicategory

A wild semicategory is a tuple $(\mathrm{Ob}, \mathrm{hom}, \circ, \alpha)$ consisting of:

- $\mathrm{Ob} : U$.
- $\mathrm{hom} : \mathrm{Ob} \to \mathrm{Ob} \to U$.
- $\_ \circ \_ : \prod_{x,y,z:\mathrm{Ob}} \mathrm{hom}\, y\ z \to \mathrm{hom}\, x\ y \to \mathrm{hom}\, x\ z$.
- $\alpha : \prod_{f,g,h} (h \circ g) \circ f = h \circ (g \circ f)$.

1. Wild semicategory

A wild semicategory is a tuple $(\mathrm{Ob}, \mathrm{hom}, \circ, \alpha)$ consisting of:

- $\mathrm{Ob} : U$.
- $\mathrm{hom} : \mathrm{Ob} \to \mathrm{Ob} \to U$.
- $\_ \circ \_ : \prod_{x,y,z:\mathrm{Ob}} \mathrm{hom}\, y\ z \to \mathrm{hom}\, x\ y \to \mathrm{hom}\, x\ z$.
- $\alpha : \prod_{f,g,h} (h \circ g) \circ f = h \circ (g \circ f)$.

Notice the lack of an identity structure.

1. Wild semicategory

A wild semicategory is a tuple $(\mathrm{Ob}, \mathrm{hom}, \circ, \alpha)$ consisting of:

- $\mathrm{Ob} : U$.
- $\mathrm{hom} : \mathrm{Ob} \to \mathrm{Ob} \to U$.
- $\_ \circ \_ : \prod_{x,y,z:\mathrm{Ob}} \mathrm{hom}\, y\ z \to \mathrm{hom}\, x\ y \to \mathrm{hom}\, x\ z$.
- $\alpha : \prod_{f,g,h} (h \circ g) \circ f = h \circ (g \circ f)$.

Notice the lack of an identity structure.

Goal of the talk:
Define an identity structure for a wild semicategory.

Preparation

We can still define notions of equivalence and neutrality using the type theoretic equivalences and identity types:

Preparation

We can still define notions of equivalence and neutrality using the type theoretic equivalences and identity types:

For x, y : Ob,

$$\mathrm{eqv}(x,y) :\equiv \sum_{f:\mathrm{hom}(x,y)} \mathrm{isequiv}(f \circ \_) \times \mathrm{isequiv}(\_ \circ f)$$

$$\mathrm{neut}(x) :\equiv \sum_{f:\mathrm{hom}(x,x)} \prod_{y:\mathrm{Ob}} (f \circ \_) = \mathrm{id}_{\mathrm{hom}(y,x)} \times \prod_{y:\mathrm{Ob}} (\_ \circ f) = \mathrm{id}_{\mathrm{hom}(x,y)}$$

Outline

2. Naive Identities

For any object x : Ob we should get

- a particular morphism $i : \hom(x, x)$.
- left and right neutrality: $\lambda_f : i \circ f = f$ and $\rho_f : f \circ i = f$.

2. Naive Identities

For any object x : Ob we should get

- a particular morphism $i : \hom(x, x)$.
- left and right neutrality: $\lambda_f : i \circ f = f$ and $\rho_f : f \circ i = f$.

Which in HoTT is written:

$$\prod_{x:\mathrm{Ob}} \sum_{i:\hom(x,x)} \prod_{y:\mathrm{Ob}} \left(\prod\nolimits_{(f:\hom(x,y))} i \circ f = f\right) \times \left(\prod\nolimits_{(f:\hom(x,y))} f \circ i = f\right)$$

2. Naive Identities

For any object $x : \mathrm{Ob}$ we should get

- a particular morphism $i : \mathrm{hom}(x, x)$.
- left and right neutrality: $\lambda_f : i \circ f = f$ and $\rho_f : f \circ i = f$.

Which in HoTT is written:

$$\prod_{x:\mathrm{Ob}} \mathrm{neut}(x)$$

2. Naive Identities

For any object $x : \mathrm{Ob}$ we should get

- a particular morphism $i : \hom(x, x)$.
- left and right neutrality: $\lambda_f : i \circ f = f$ and $\rho_f : f \circ i = f$.

Which in HoTT is written:

$$\prod_{x:\mathrm{Ob}} \mathrm{neut}(x)$$

This is not a proposition, we need the propositional truncation.

Naive Identities

$$\mathrm{NaId} :\equiv \prod_{x:\mathrm{Ob}} \|\,\mathrm{neut}(x)\|$$

Outline

1. Wild semicategory
2. Naive Identities
3. (Co)slice approach
4. Harpaz's identities
5. Idempotent Equivalences
6. Comparison

3. (Co)slice approach

Reminder

For an object x in a category C, the slice category C/x (resp. coslice category $x\backslash C$) consists of morphisms $a \to x$ (resp. $x \to a$) as objects and of commutative triangles as morphisms.

$$\begin{array}{ccc} a & \longrightarrow & b \\ & \searrow \quad \swarrow & \\ & x & \end{array} \qquad \left(\text{resp. } \begin{array}{ccc} a & \longrightarrow & b \\ & \nwarrow \quad \nearrow & \\ & x & \end{array} \right)$$

3. (Co)slice approach

Reminder

For an object x in a category C, the slice category C/x (resp. coslice category $x\backslash C$) consists of morphisms $a \to x$ (resp. $x \to a$) as objects and of commutative triangles as morphisms.

$$\begin{array}{ccc} a & \longrightarrow & b \\ & \searrow \quad \swarrow & \\ & x & \end{array} \qquad\qquad \left(\text{resp.} \begin{array}{ccc} a & \longrightarrow & b \\ & \nwarrow \quad \nearrow & \\ & x & \end{array} \right)$$

Remark

In the slice category C/x the identity $\mathrm{id}_x : x \to x$ is terminal.
Dually, id_x is initial in $x\backslash C$.

$\rightsquigarrow$ Use this fact to define the identity structure in a wild semicategory.

3. (Co)slice approach

A straightforward way to express the structure is thus to ask for all slices to have a terminal object and all coslices to have an initial object:

3. (Co)slice approach

A straightforward way to express the structure is thus to ask for all slices to have a terminal object and all coslices to have an initial object:

$$\prod_{x:\mathsf{Ob}} \mathsf{hasTerm}(C/x) \times \mathsf{hasInit}(x \backslash C)$$

3. (Co)slice approach

A straightforward way to express the structure is thus to ask for all slices to have a terminal object and all coslices to have an initial object:

$$\prod_{x:\mathrm{Ob}} \sum_{t:\mathrm{hom}(\_,x)} \mathrm{isequiv}(t \circ \_) \times \sum_{i:\mathrm{hom}(x,\_)} \mathrm{isequiv}(\_ \circ i)$$

3. (Co)slice approach

A straightforward way to express the structure is thus to ask for all slices to have a terminal object and all coslices to have an initial object:

$$\prod_{x:\mathsf{Ob}} \sum_{t:\hom(\_,x)} \mathsf{isequiv}(t \circ \_) \times \sum_{i:\hom(x,\_)} \mathsf{isequiv}(\_ \circ i)$$

Issues:

- i and t might not have same (co)domain.
- $\hom(x, x)$ might be empty.

$\rightsquigarrow \mathbb{Z}$ as a semicategory is a counterexample.

3. (Co)slice approach

A straightforward way to express the structure is thus to ask for all slices to have a terminal object and all coslices to have an initial object:

$$\prod_{x:\mathrm{Ob}} \sum_{t:\hom(\_,x)} \mathrm{isequiv}(t \circ \_) \times \sum_{i:\hom(x,\_)} \mathrm{isequiv}(\_ \circ i)$$

Issues:

- i and t might not have same (co)domain.
- $\hom(x,x)$ might be empty.

$\rightsquigarrow$ $\mathbb{Z}$ as a semicategory is a counterexample.

Identity structure using (co)slice approach

$$\mathrm{SliceId} :\equiv \prod_{x:\mathrm{Ob}} \exists_{i:\hom(x,x)} \mathrm{isequiv}(i \circ \_) \times \mathrm{isequiv}(\_ \circ i)$$

3. (Co)slice approach

A straightforward way to express the structure is thus to ask for all slices to have a terminal object and all coslices to have an initial object:

$$\prod_{x:\mathsf{Ob}} \sum_{t:\mathsf{hom}(\_,x)} \mathsf{isequiv}(t \circ \_) \times \sum_{i:\mathsf{hom}(x,\_)} \mathsf{isequiv}(\_ \circ i)$$

Issues:

- i and t might not have same (co)domain.
- $\mathsf{hom}(x, x)$ might be empty.

$\rightsquigarrow$ $\mathbb{Z}$ as a semicategory is a counterexample.

Identity structure using (co)slice approach

$$\mathsf{SliceId} :\equiv \prod_{x:\mathsf{Ob}} \|\mathsf{eqv}(x, x)\|$$

Outline

4. Harpaz's identities

Harpaz[1] gives a similar identity structure but weaker:

Harpaz's Identities

$$\mathsf{HarpazId} :\equiv \prod_{x:\mathsf{Ob}} \exists_{y:\mathsf{Ob}} \mathsf{eqv}(x, y)$$

[1] Y. Harpaz, Quasi-unital ∞-categories (2015)

4. Harpaz's identities

Harpaz[1] gives a similar identity structure but weaker:

Harpaz's Identities

$$\mathsf{HarpazId} :\equiv \prod_{x:\mathsf{Ob}} \exists_{y:\mathsf{Ob}} \mathsf{eqv}(x, y)$$

Which can easily be made into an univalent identity structure:

Univalent Harpaz's Identities

$$\mathsf{uHarpazId} :\equiv \prod_{x:\mathsf{Ob}} \mathsf{isContr}\left(\sum_{y:\mathsf{Ob}} \mathsf{eqv}(x, y)\right)$$

[1] Y. Harpaz, Quasi-unital ∞-categories (2015)

Outline

5. Idempotent Equivalences

So far, we always used propositional truncation.

Question

Can we avoid the use of propositional truncation?

5. Idempotent Equivalences

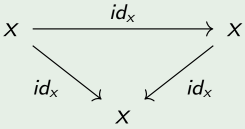
So far, we always used propositional truncation.

Question

Can we avoid the use of propositional truncation?

Remark

In C/x and $x\backslash C$ the only endomorphism of the identity is the identity itself:



5. Idempotent Equivalences

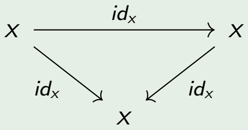
So far, we always used propositional truncation.

Question

Can we avoid the use of propositional truncation?

Remark

In C/x and $x\backslash C$ the only endomorphism of the identity is the identity itself:



For a morphism f, the property $f \circ f = f$ is called idempotency.

5. Idempotent Equivalences

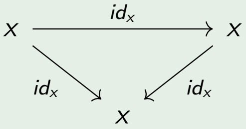
So far, we always used propositional truncation.

Question

Can we avoid the use of propositional truncation?

Remark

In C/x and $x\backslash C$ the only endomorphism of the identity is the identity itself:



For a morphism f, the property $f \circ f = f$ is called idempotency.

Hence, identity-like morphisms should have this property.

5. Idempotent Equivalences

This kind of identity structure have been studied by Saavera[2] and Kock[3]. Nicolai showed in his paper[4] that idempotency is actually enough:

[2] N. Saavera Rivano, Catégories Tannakiennes (1972)

[3] J. Kock, Elementary remarks on units in monoidal categories (2008)

[4] N. Kraus, Internal ∞-categorical models of dependent type theory: Towards 2LTT eating HoTT (2021)

5. Idempotent Equivalences

This kind of identity structure have been studied by Saavera[2] and Kock[3]. Nicolai showed in his paper[4] that idempotency is actually enough:

Proposition (Kraus, 2021)

The type

$$\mathsf{IdemEqv} :\equiv \prod_{x:\mathsf{Ob}} \sum_{i:\mathsf{eqv}(x,x)} i \circ i = i$$

is a proposition.

[2] N. Saavera Rivano, Catégories Tannakiennes (1972)

[3] J. Kock, Elementary remarks on units in monoidal categories (2008)

[4] N. Kraus, Internal ∞-categorical models of dependent type theory: Towards 2LTT eating HoTT (2021)

5. Idempotent Equivalences

This kind of identity structure have been studied by Saavera[2] and Kock[3]. Nicolai showed in his paper[4] that idempotency is actually enough:

Proposition (Kraus, 2021)

The type

$$\mathsf{IdemEqv} :\equiv \prod_{x:\mathsf{Ob}} \sum_{i:\mathsf{eqv}(x,x)} i \circ i = i$$

is a proposition.

Hence we can take it as identity structure on a wild semicategory.

- No truncation needed.
- We can project out the identity morphism and easily define left and right neutrality (λ and ρ).

[2] N. Saavera Rivano, Catégories Tannakiennes (1972)

[3] J. Kock, Elementary remarks on units in monoidal categories (2008)

[4] N. Kraus, Internal ∞-categorical models of dependent type theory: Towards 2LTT eating HoTT (2021)

Outline

6. Comparison

Theorem

For a given wild semicategory, the four types NaId, IdemEqv, HarpazId, SliceId *are equivalent propositions.*

6. Comparison

Theorem

For a given wild semicategory, the four types NaId, IdemEqv, HarpazId, SliceId *are equivalent propositions.*

Proof.

- The equivalence NaId $\Leftrightarrow$ IdemEqv is straightforward.
- HarpazId $\Rightarrow$ IdemEqv amounts to showing that for an equivalence $x \xrightarrow{f} y$ the morphisms $(f \circ \_)^{-1}(f)$ is an idempotent equivalence.
- IdemEqv $\Rightarrow$ SliceId is the projection.
- SliceId $\Rightarrow$ HarpazId is trivial. □

Conclusion

Overview

Id Structure	Avoids truncation	Captures non-univalent categories
NaId	✗	✓
SliceId	✗	✓
HarpazId	✗	✓
uHarpazId	✓	✗
IdemEqv	✓	✓

Conclusion

Overview

Id Structure	Avoids truncation	Captures non-univalent categories
NaId	✗	✓
SliceId	✗	✓
HarpazId	✗	✓
uHarpazId	✓	✗
IdemEqv	✓	✓

Conjecture

Low levels are enough to produce all the higher coherences for the identity structure.

Conclusion

Overview

Id Structure	Avoids truncation	Captures non-univalent categories
NaId	✗	✓
SliceId	✗	✓
HarpazId	✗	✓
uHarpazId	✓	✗
IdemEqv	✓	✓

Conjecture

Low levels are enough to produce all the higher coherences for the identity structure.

Formalisation in Agda:
See `github.com/jaycech3n/semicategories-with-identities` for details.