Invertibility in higher categories

Ioannis Markakis

University of Cambridge

12-14 August 2026

The need for higher categories

Many concepts in mathematics are not modelled well by ordinary categories, but naturally live within higher dimensional structures:

The need for higher categories

Many concepts in mathematics are not modelled well by ordinary categories, but naturally live within higher dimensional structures:

1. Categories form a strict 2-category.
   Strict 2-categories form a strict 3-category.
   Strict n-categories form a strict $(n+1)$-category.

The need for higher categories

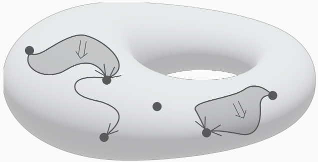
Many concepts in mathematics are not modelled well by ordinary categories, but naturally live within higher dimensional structures:

1. Categories form a strict 2-category.
   Strict 2-categories form a strict 3-category.
   Strict n-categories form a strict $(n+1)$-category.
2. Every monoidal category is a bicategory with one object.
   Rings with bimodules form a bicategory or a double category.
   Categories with profunctors form a bicategory.

The need for higher categories

Many concepts in mathematics are not modelled well by ordinary categories, but naturally live within higher dimensional structures:

1. Categories form a strict 2-category.
   Strict 2-categories form a strict 3-category.
   Strict n-categories form a strict $(n+1)$-category.
2. Every monoidal category is a bicategory with one object.
   Rings with bimodules form a bicategory or a double category.
   Categories with profunctors form a bicategory.
3. Each topological space X forms an ∞-groupoid $\Pi_\infty(X)$ with:
   - objects the points of X,
   - 1-morphisms the paths in X,
   - 2-morphisms the path homotopies,
   - higher morphisms the homotopies between homotopies.



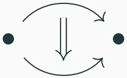
Variants of higher structures

Models for higher categories are roughly classified by:

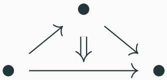
Variants of higher structures

Models for higher categories are roughly classified by:

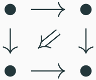
- the underlying shapes of their higher morphisms:



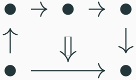
Globular



Simplicial



Cubical



Opetopic

Variants of higher structures

Models for higher categories are roughly classified by:

- the underlying shapes of their higher morphisms:

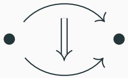
Globular **Simplicial** **Cubical** **Opetopic**

- their cell dimension n and their invertibility dimension k

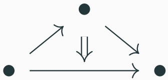
Variants of higher structures

Models for higher categories are roughly classified by:

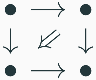
- the underlying shapes of their higher morphisms:



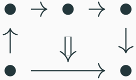
Globular



Simplicial



Cubical



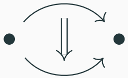
Opetopic

- their cell dimension n and their invertibility dimension k
- whether composition is property or structure,

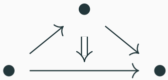
Variants of higher structures

Models for higher categories are roughly classified by:

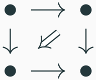
- the underlying shapes of their higher morphisms:



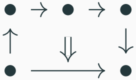
Globular



Simplicial



Cubical



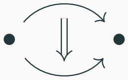
Opetopic

- their cell dimension n and their invertibility dimension k
- whether composition is property or structure,
- whether composition is strict or weak.

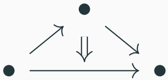
Variants of higher structures

Models for higher categories are roughly classified by:

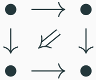
- the underlying shapes of their higher morphisms:



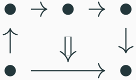
Globular



Simplicial



Cubical



Opetopic

- their cell dimension n and their invertibility dimension k
- whether composition is property or structure,
- whether composition is strict or weak.

Remark

We will focus on algebraic globular weak (∞, ∞)-categories, henceforth ω-categories.

Invertibility in higher categories

Invertible morphisms allow us to define equivalence within and between higher categories.

Invertibility in higher categories

Invertible morphisms allow us to define equivalence within and between higher categories.

- A morphism $f\colon x \to y$ in an 1-category is invertible if there exist $f^l, f^r\colon y \to x$ such that

$$f \circ f^l = 1(x) \qquad\qquad 1(y) = f^r \circ f$$

Invertibility in higher categories

Invertible morphisms allow us to define equivalence within and between higher categories.

- A morphism $f\colon x \to y$ in an 1-category is invertible if there exist $f^l, f^r\colon y \to x$ such that
$$f \circ f^l = 1(x) \qquad\qquad 1(y) = f^r \circ f$$
- A morphism $f\colon x \to y$ in an $(n+1)$-category is invertible if there exist $f^l, f^r\colon y \to x$ and invertible
$$\epsilon_f\colon f \circ f^l \Rightarrow 1(x) \qquad\qquad \eta_f\colon 1(y) \Rightarrow f^r \circ f$$
in the n-categories $\mathrm{Hom}(x,x)$ and $\mathrm{Hom}(y,y)$.

$$\begin{tikzcd}
y \arrow[rr, equal] \arrow[dr, "f^l"'] & & y \arrow[dr, "f^r"] \arrow[d, phantom, "\wr\Uparrow\eta_f"] & \\
& x \arrow[ur, "f" description] \arrow[rr, equal] \arrow[u, phantom, "\wr\Uparrow\epsilon_f"] & {} & x
\end{tikzcd}$$

Invertibility in higher categories

Invertible morphisms allow us to define equivalence within and between higher categories.

- A morphism $f\colon x \to y$ in an 1-category is invertible if there exist $f^l, f^r\colon y \to x$ such that

$$f \circ f^l = 1(x) \qquad\qquad 1(y) = f^r \circ f$$

- A morphism $f\colon x \to y$ in an $(n+1)$-category is invertible if there exist $f^l, f^r\colon y \to x$ and invertible

$$\epsilon_f\colon f \circ f^l \Rightarrow 1(x) \qquad\qquad \eta_f\colon 1(y) \Rightarrow f^r \circ f$$

  in the n-categories $\mathrm{Hom}(x,x)$ and $\mathrm{Hom}(y,y)$.

$$\begin{tikzcd}
y \arrow[rr, equal] \arrow[dr, "f^l"'] & & y \arrow[dr, "f^r"] \arrow[d, phantom, "\wr\Uparrow\eta_f"] & \\
& x \arrow[ur, "f"] \arrow[rr, equal] \arrow[u, phantom, "\wr\Uparrow\epsilon_f"] & {} & x
\end{tikzcd}$$

Definition

For ω-categories this definition is made coinductive: invertibility of a morphism is witnessed by an infinite tower of invertible higher morphisms.

Objectives of this talk

1. What is an ω-category?

2. What can we freely generate ω-categories from?

3. Can we make some of the generating morphisms as invertible?

Objectives of this talk

1. What is an ω-category?
   - Batanin [3] and Leinster [13] define them as using globular operads
   - Grothendieck [11] and Maltsiniotis [14] define them using limit sketchs
   - Finster and Mimram [10] define them using a dependent type theory
2. What can we freely generate ω-categories from?
3. Can we make some of the generating morphisms as invertible?

Objectives of this talk

1. What is an ω-category?
   - Batanin [3] and Leinster [13] define them as using globular operads
   - Grothendieck [11] and Maltsiniotis [14] define them using limit sketchs
   - Finster and Mimram [10] define them using a dependent type theory
2. What can we freely generate ω-categories from?
   - Globular sets is an easy answer, but not the most general one
   - Batanin [2] and Burroni [8] use computads (a.k.a. polygraphs)
   - Dean, Finster, M., Reutter and Vicary [9] introduce an inductive presentation ω-categories freely generated from a computad
3. Can we make some of the generating morphisms as invertible?

Objectives of this talk

1. What is an ω-category?
   - Batanin [3] and Leinster [13] define them as using globular operads
   - Grothendieck [11] and Maltsiniotis [14] define them using limit sketchs
   - Finster and Mimram [10] define them using a dependent type theory
2. What can we freely generate ω-categories from?
   - Globular sets is an easy answer, but not the most general one
   - Batanin [2] and Burroni [8] use computads (a.k.a. polygraphs)
   - Dean, Finster, M., Reutter and Vicary [9] introduce an inductive presentation ω-categories freely generated from a computad
3. Can we make some of the generating morphisms as invertible?
   - Benjamin, Champin, M. [4] generalise computads to allow inverting generators

Objectives of this talk

1. What is an ω-category?
   - Batanin [3] and Leinster [13] define them as using globular operads
   - Grothendieck [11] and Maltsiniotis [14] define them using limit sketchs
   - Finster and Mimram [10] define them using a dependent type theory
2. What can we freely generate ω-categories from?
   - Globular sets is an easy answer, but not the most general one
   - Batanin [2] and Burroni [8] use computads (a.k.a. polygraphs)
   - Dean, Finster, M., Reutter and Vicary [9] introduce an inductive presentation ω-categories freely generated from a computad
3. Can we make some of the generating morphisms as invertible?
   - Benjamin, Champin, M. [4] generalise computads to allow inverting generators

Remark

All definitions of ω-categories have been shown equivalent by Ara [1], Bourke [7], Benjamin, Finster, Mimram [6], and Benjamin, M., Sarti [5].

Strict ω-categories i

Definition

Strict 0-categories are sets.
Strict $(n+1)$-categories are categories enriched in strict n-categories.
The category of strict ω-categories is the limit of the categories of strict n-categories.

Strict ω-categories i

Definition

Strict 0-categories are sets.
Strict $(n+1)$-categories are categories enriched in strict n-categories.
The category of strict ω-categories is the limit of the categories of strict n-categories.

Pros

- They admit a simple inductive definition.
- Their homotopy theory is understood [12].

Cons

- Most examples of higher categories are not strict.
- Not all tricategories are equivalent to strict ones.

Globular sets

Definition

The category $\mathbb{G}$ of globes is the category generated by the following graph subject to the globularity relations:

$$0 \overset{\sigma_0}{\underset{\tau_0}{\rightrightarrows}} 1 \overset{\sigma_1}{\underset{\tau_1}{\rightrightarrows}} 2 \overset{\sigma_2}{\underset{\tau_2}{\rightrightarrows}} \dots \qquad \begin{aligned} \sigma_{n+1} \circ \sigma_n &= \tau_{n+1} \circ \sigma_n \\ \sigma_{n+1} \circ \tau_n &= \tau_{n+1} \circ \tau_n \end{aligned}$$

The category Glob of globular sets is the presheaf category $[\mathbb{G}^{\mathrm{op}}, \mathrm{Set}]$.

Globular sets

Definition

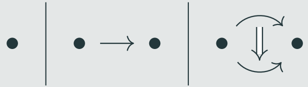
The category $\mathbb{G}$ of globes is the category generated by the following graph subject to the globularity relations:

$$
\begin{tikzcd} 0 \arrow[r, shift left, "\sigma_0"] \arrow[r, shift right, "\tau_0"'] & 1 \arrow[r, shift left, "\sigma_1"] \arrow[r, shift right, "\tau_1"'] & 2 \arrow[r, shift left, "\sigma_2"] \arrow[r, shift right, "\tau_2"'] & \dots \end{tikzcd}
\qquad
\begin{aligned} \sigma_{n+1} \circ \sigma_n &= \tau_{n+1} \circ \sigma_n \\ \sigma_{n+1} \circ \tau_n &= \tau_{n+1} \circ \tau_n \end{aligned}
$$

The category Glob of globular sets is the presheaf category $[\mathbb{G}^{\mathrm{op}}, \mathrm{Set}]$.

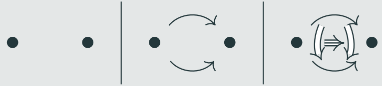
Example

The disks $\mathbb{D}^n = \mathbb{G}(-, n)$ are the representable globular sets.



Example

The spheres $\mathbb{S}^n = \mathbb{D}^{n+1} \setminus \{\mathrm{id}_{n+1}\}$ are their boundaries.



Strict ω-categories ii

Definition

A strict ω-category is a globular set X equipped with identity and composition operations for $k < n$:

$$\circ_k \colon X_n \times_{X_k} X_n \to X_n \qquad\qquad 1_n \colon X_n \to X_{n+1}$$

satisfying associativity, unitality and interchange laws.

Definition

A strict ω-category is a globular set X equipped with identity and composition operations for $k < n$:

$$\circ_k \colon X_n \times_{X_k} X_n \to X_n \qquad\qquad 1_n \colon X_n \to X_{n+1}$$

satisfying associativity, unitality and interchange laws.

$$\overset{x}{\bullet} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, "1_0(x)"] & \overset{x}{\bullet} \end{tikzcd}$$

$$\begin{tikzcd} \overset{x}{\bullet} \arrow[r, "f"] & \overset{y}{\bullet} \arrow[r, "g"] & \overset{z}{\bullet} \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, "f \circ_0 g"] & \overset{z}{\bullet} \end{tikzcd}$$

$$\begin{tikzcd} \overset{x}{\bullet} \arrow[r, bend left=60, "f"] \arrow[r, "g" description] \arrow[r, bend right=60, "h"'] \arrow[r, phantom, shift left=3, "\Downarrow\alpha"] \arrow[r, phantom, shift right=3, "\Downarrow\beta"] & \overset{y}{\bullet} \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, bend left=60, "f"] \arrow[r, bend right=60, "h"'] \arrow[r, phantom, "\Downarrow\alpha \circ_1 \beta"] & \overset{y}{\bullet} \end{tikzcd}$$

$$\begin{tikzcd} \overset{x}{\bullet} \arrow[r, bend left=50, "f"] \arrow[r, bend right=50, "g"'] \arrow[r, phantom, "\Downarrow\alpha"] & \overset{y}{\bullet} \arrow[r, bend left=50, "h"] \arrow[r, bend right=50, "k"'] \arrow[r, phantom, "\Downarrow\beta"] & \overset{z}{\bullet} \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, bend left=60, "h \circ_0 f"] \arrow[r, bend right=60, "k \circ_0 g"'] \arrow[r, phantom, "\Downarrow\alpha \circ_1 \beta"] & \overset{z}{\bullet} \end{tikzcd}$$

Pasting diagrams

Pasting diagrams are a family of globular sets that admit a unique composite in any strict ω-category. They are generated inductively from $\mathbb{D}^0$ using the suspension and wedge sum operations:

Pasting diagrams

Pasting diagrams are a family of globular sets that admit a unique composite in any strict ω-category. They are generated inductively from $\mathbb{D}^0$ using the suspension and wedge sum operations:

Definition

The suspension of a globular set X has two objects u, v with $\mathrm{Hom}(u, v) = X$ and all other homs empty.

$$x \xrightarrow{f} y \xrightarrow{g} z \quad \rightsquigarrow \quad \begin{tikzcd} u \arrow[d, bend right=80, "x"'{name=X}] \arrow[d, "y"{name=Y}] \arrow[d, bend left=80, "z"{name=Z}] \arrow[from=X, to=Y, Rightarrow, "f"] \arrow[from=Y, to=Z, Rightarrow, "g"] \\ v \end{tikzcd}$$

Pasting diagrams

Pasting diagrams are a family of globular sets that admit a unique composite in any strict ω-category. They are generated inductively from $\mathbb{D}^0$ using the suspension and wedge sum operations:

Definition

The suspension of a globular set X has two objects u, v with $\mathrm{Hom}(u, v) = X$ and all other homs empty.

$$x \xrightarrow{f} y \xrightarrow{g} z \quad \rightsquigarrow \quad \begin{tikzcd} u \arrow[d, bend right=60, "x"'] \arrow[d, "y" description] \arrow[d, bend left=60, "z"] \arrow[d, phantom, shift right=4, "\xRightarrow{f}"] \arrow[d, phantom, shift left=4, "\xRightarrow{g}"] \\ v \end{tikzcd}$$

Definition

The wedge sum of globular sets X and Y with chosen objects $x, x' \in X_0$ and $y, y' \in Y_0$ is the pushout:

$$\begin{tikzcd} & & X \vee Y \arrow[dd, phantom, "\vee" very near start] & & \\ & X \arrow[ur, dashed] & & Y \arrow[ul, dashed] & \\ \mathbb{D}^0 \arrow[ur, "x"] & & \mathbb{D}^0 \arrow[ul, "x'"'] \arrow[ur, "y"] & & \mathbb{D}^0 \arrow[ul, "y'"'] \end{tikzcd}$$

Batanin trees

Batanin [3] showed that pasting diagrams are parametrised by rooted, planar trees:

Definition

The set Bat of Batanin trees is defined inductively by br L being a tree for any list L of Batanin trees.

Batanin trees

Batanin [3] showed that pasting diagrams are parametrised by rooted, planar trees:

Definition

The set Bat of Batanin trees is defined inductively by $\mathrm{br}\, L$ being a tree for any list L of Batanin trees. The pasting diagram associated to a tree is defined recursively by

$$\mathrm{Pos}(\mathrm{br}[B_1, \ldots, B_n]) = \bigvee_{i=1}^{n} \Sigma \, \mathrm{Pos}(B_i)$$

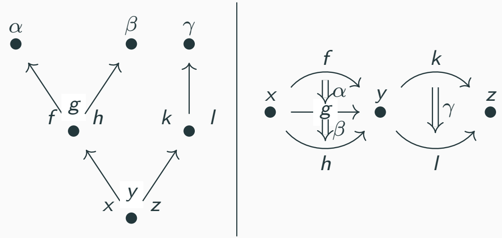
Batanin trees

Batanin [3] showed that pasting diagrams are parametrised by rooted, planar trees:

Definition

The set Bat of Batanin trees is defined inductively by $\mathrm{br}\, L$ being a tree for any list L of Batanin trees. The pasting diagram associated to a tree is defined recursively by

$$\mathrm{Pos}(\mathrm{br}[B_1, \ldots, B_n]) = \bigvee_{i=1}^{n} \Sigma\, \mathrm{Pos}(B_i)$$



$$B = \mathrm{br}[\mathrm{br}[\mathrm{br}[], \mathrm{br}[]], \mathrm{br}[\mathrm{br}[]]]$$

Towards weak ω-categories

Guiding principles

To define weak ω-categories, we replace uniqueness with uniqueness up to equivalence:

Towards weak ω-categories

Guiding principles

To define weak ω-categories, we replace uniqueness with uniqueness up to equivalence:

- Parallel ways to compose the source and target of a pasting diagram give rise to a way to compose the entire pasting diagram.
- Parallel composites or coherences over a pasting diagram are related by a coherence operation over the same pasting diagram.

$$\begin{tikzcd} \overset{x}{\bullet} \arrow[r, "f"] & \overset{y}{\bullet} \arrow[r, "g"] & \overset{z}{\bullet} \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, "f \circ g"] & \overset{y}{\bullet} \end{tikzcd} \qquad \Bigg| \qquad \overset{x}{\bullet} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, "1(x)"] & \overset{x}{\bullet} \end{tikzcd}$$

$$\begin{tikzcd} \overset{x}{\bullet} \arrow[r, "f"] & \overset{y}{\bullet} \arrow[r, "g"] & \overset{z}{\bullet} \arrow[r, "h"] & \overset{w}{\bullet} \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, bend left, "(f \circ g) \circ h"] \arrow[r, bend right, "f \circ (g \circ h)"'] \arrow[r, phantom, "\Downarrow \alpha_{f,g,h}"] & \overset{w}{\bullet} \end{tikzcd} \qquad \Bigg| \qquad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, "f"] & \overset{y}{\bullet} \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} \overset{x}{\bullet} \arrow[r, bend left, "f \circ 1(y)"] \arrow[r, bend right, "f"'] \arrow[r, phantom, "\Downarrow \rho_f"] & \overset{x}{\bullet} \end{tikzcd}$$

Strict ω-categories iii

Definition

Strict ω-categories are algebras for a monad T^{str} on Glob:

$$(T^{\mathrm{str}}X)_n = \coprod_{\dim B \leq n} \mathrm{Glob}(\mathrm{Pos}(B), X)$$

Strict ω-categories iii

Definition

Strict ω-categories are algebras for a monad T^{str} on Glob:

$$(T^{\mathrm{str}}X)_n = \coprod_{\dim B \leq n} \mathrm{Glob}(\mathrm{Pos}(B), X)$$

It follows that there is an adjunction:

$$\begin{tikzcd} \mathrm{Glob} \arrow[r, shift left=2, "F^{\mathrm{str}}"] \arrow[r, phantom, "\perp"] & \mathrm{Cat}_\omega \arrow[l, shift left=2, "U^{\mathrm{str}}"] \end{tikzcd}$$

so we can generate strict ω-categories from globular sets.

Strict ω-categories iii

Definition

Strict ω-categories are algebras for a monad T^{str} on Glob:

$$(T^{\mathrm{str}}X)_n = \coprod_{\dim B \le n} \mathrm{Glob}(\mathrm{Pos}(B), X)$$

It follows that there is an adjunction:

$$\begin{tikzcd} \mathrm{Glob} \arrow[r, shift left=1.5ex, "F^{\mathrm{str}}"] \arrow[r, phantom, "\perp"] & \mathrm{Cat}_\omega \arrow[l, shift left=1.5ex, "U^{\mathrm{str}}"] \end{tikzcd}$$

so we can generate strict ω-categories from globular sets.

Question

What about more general shapes of generating data:

- simplices, cubes, opetopes...
- the walking isomorphism $x \xrightarrow{\sim} y$
- the walking commmutative square

Street [15] defined computads as ways to generate 2-categories. Burroni [8] generalised them to generate strict ω-categories. Batanin [2] further generalised them to generate algebras for finitary monads on globular sets, including strict and weak ω-categories.

Idea

An n-computad is a diagram of sets of the following form

$$\begin{tikzcd}
V_0^C \arrow[d] & V_1^C \arrow[d] \arrow[dl, shift left] \arrow[dl, shift right] & V_2^C \arrow[d] \arrow[dl, shift left] \arrow[dl, shift right] & \dots \arrow[dl, shift left] \arrow[dl, shift right] & V_n^C \arrow[dl, shift left] \arrow[dl, shift right] \\
TC_0 & TC_1 \arrow[l, shift left] \arrow[l, shift right] & TC_2 \arrow[l, shift left] \arrow[l, shift right] & \dots \arrow[l, shift left] \arrow[l, shift right] &
\end{tikzcd}$$

where TC_k is the T-algebra generated by the k-computad $(V_0^C, \dots, V_k^C)$.

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathrm{tr}_n \colon \mathrm{Comp}_{n+1} \to \mathrm{Comp}_n$

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathrm{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathrm{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathsf{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathsf{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$
4. two copresheaves $\mathsf{Cell}_n, \mathsf{Sphere}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathsf{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathsf{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$
4. two copresheaves $\mathsf{Cell}_n, \mathsf{Sphere}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
5. three natural transformations $\mathsf{bdry}_n \colon \mathsf{Cell}_{n+1} \Rightarrow \mathsf{Sphere}_n \mathsf{tr}_n$ and $\mathsf{pr}_{1,n}, \mathsf{pr}_{2,n} \colon \mathsf{Sphere}_n \Rightarrow \mathsf{Cell}_n$

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathrm{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathrm{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$
4. two copresheaves $\mathrm{Cell}_n, \mathrm{Sphere}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
5. three natural transformations $\mathrm{bdry}_n \colon \mathrm{Cell}_{n+1} \Rightarrow \mathrm{Sphere}_n \, \mathrm{tr}_n$ and $\mathrm{pr}_{1,n}, \mathrm{pr}_{2,n} \colon \mathrm{Sphere}_n \Rightarrow \mathrm{Cell}_n$
6. A subset $\mathrm{Full}_n(B) \subseteq \mathrm{Sphere}_n(\mathrm{Cptd}_n \, \mathrm{Pos}(B))$ for each Batanin tree B

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathsf{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathsf{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$
4. two copresheaves $\mathsf{Cell}_n, \mathsf{Sphere}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
5. three natural transformations $\mathsf{bdry}_n \colon \mathsf{Cell}_{n+1} \Rightarrow \mathsf{Sphere}_n \, \mathsf{tr}_n$ and $\mathsf{pr}_{1,n}, \mathsf{pr}_{2,n} \colon \mathsf{Sphere}_n \Rightarrow \mathsf{Cell}_n$
6. A subset $\mathsf{Full}_n(B) \subseteq \mathsf{Sphere}_n(\mathsf{Cptd}_n \, \mathsf{Pos}(B))$ for each Batanin tree B
7. a functor $\mathsf{Mark}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathsf{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathsf{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$
4. two copresheaves $\mathsf{Cell}_n, \mathsf{Sphere}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
5. three natural transformations $\mathsf{bdry}_n \colon \mathsf{Cell}_{n+1} \Rightarrow \mathsf{Sphere}_n \, \mathsf{tr}_n$ and $\mathsf{pr}_{1,n}, \mathsf{pr}_{2,n} \colon \mathsf{Sphere}_n \Rightarrow \mathsf{Cell}_n$
6. A subset $\mathsf{Full}_n(B) \subseteq \mathsf{Sphere}_n(\mathsf{Cptd}_n \, \mathsf{Pos}(B))$ for each Batanin tree B
7. a functor $\mathsf{Mark}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
8. natural transformations $\mathsf{fwd}_n, \mathsf{linv}_n, \mathsf{rinv}_n \colon \mathsf{Mark}_n \Rightarrow \mathsf{Cell}_n$

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathsf{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathsf{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$
4. two copresheaves $\mathsf{Cell}_n, \mathsf{Sphere}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
5. three natural transformations $\mathsf{bdry}_n \colon \mathsf{Cell}_{n+1} \Rightarrow \mathsf{Sphere}_n \mathsf{tr}_n$ and $\mathsf{pr}_{1,n}, \mathsf{pr}_{2,n} \colon \mathsf{Sphere}_n \Rightarrow \mathsf{Cell}_n$
6. A subset $\mathsf{Full}_n(B) \subseteq \mathsf{Sphere}_n(\mathsf{Cptd}_n \mathsf{Pos}(B))$ for each Batanin tree B
7. a functor $\mathsf{Mark}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
8. natural transformations $\mathsf{fwd}_n, \mathsf{linv}_n, \mathsf{rinv}_n \colon \mathsf{Mark}_n \Rightarrow \mathsf{Cell}_n$

The formal specification

Our goal is to define weak ω-categories and their computads.
To that end we define mutually recursively on the dimension n:

1. a category Comp_n of computads with invertible generators
2. a truncation functor $\mathsf{tr}_n \colon \mathsf{Comp}_{n+1} \to \mathsf{Comp}_n$
3. an inclusion functor $\mathsf{Cptd}_n \colon \mathsf{Glob} \to \mathsf{Comp}_n$
4. two copresheaves $\mathsf{Cell}_n, \mathsf{Sphere}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
5. three natural transformations $\mathsf{bdry}_n \colon \mathsf{Cell}_{n+1} \Rightarrow \mathsf{Sphere}_n \mathsf{tr}_n$ and $\mathsf{pr}_{1,n}, \mathsf{pr}_{2,n} \colon \mathsf{Sphere}_n \Rightarrow \mathsf{Cell}_n$
6. A subset $\mathsf{Full}_n(B) \subseteq \mathsf{Sphere}_n(\mathsf{Cptd}_n \mathsf{Pos}(B))$ for each Batanin tree B
7. a functor $\mathsf{Mark}_n \colon \mathsf{Comp}_n \to \mathsf{Set}$
8. natural transformations $\mathsf{fwd}_n, \mathsf{linv}_n, \mathsf{rinv}_n \colon \mathsf{Mark}_n \Rightarrow \mathsf{Cell}_n$

starting from $\mathsf{Comp}_{-1} = \mathbb{1}$, $\mathsf{Sphere}_{-1} = \{\star\}$ and $\mathsf{Cell}_{-1} = \mathsf{Mark}_{-1} = \emptyset$.

Computads and their morphisms

Definition

An $(n+1)$-computad C with invertible generators consists of:

- an n-computad $\operatorname{tr}_n C$
- two sets V^C_{n+1} and U^C_{n+1} of ordinary and invertible generators
- a function $\phi^C_n \colon V^C_{n+1} + U^C_{n+1} \to \mathrm{Sphere}_n(C_n)$ giving the boundary of each generator

Definition

A morphism $f \colon C \to D$ of $(n+1)$-computads consists of:

- a morphism $\operatorname{tr}_n f \colon \operatorname{tr}_n C \to \operatorname{tr}_n D$
- a function $f_V \colon V^C_{n+1} \to \mathrm{Cell}_{n+1}(D)$
- a function $f_U \colon U^C_{n+1} \to \mathrm{Mark}_{n+1}(D)$

$$\begin{tikzcd} V^C_{n+1} + U^C_{n+1} \arrow[r, "\langle f_V, \mathrm{fwd} \circ f_U \rangle"] \arrow[d, "\phi^C_{n+1}"] & \mathrm{Cell}_{n+1}(D) \arrow[d, "\mathrm{bdry}"] \\ \mathrm{Sphere}_n(\operatorname{tr}_n C) \arrow[r, "\mathrm{Sphere}_n(\operatorname{tr}_n f)"] & \mathrm{Sphere}_n(\operatorname{tr}_n D) \end{tikzcd}$$

Cells and their boundaries

Definition

The set $\mathrm{Mark}_{n+1}(C)$ is inductively generated by:

- $\mathrm{igen}(u)$ for $u \in U^C_{n+1}$
- $\mathrm{lunit}(i)$ and $\mathrm{runit}(i)$ for $i \in \mathrm{Mark}_n(C)$

Cells and their boundaries

Definition

The set $\mathrm{Mark}_{n+1}(C)$ is inductively generated by:

- $\mathrm{igen}(u)$ for $u \in U^C_{n+1}$
- $\mathrm{lunit}(i)$ and $\mathrm{runit}(i)$ for $i \in \mathrm{Mark}_n(C)$

Definition

The set $\mathrm{Cell}_{n+1}(C)$ is inductively generated by:

- $\mathrm{gen}(v)$ for $v \in V^C_{n+1}$
- $\mathrm{fwd}(i)$, $\mathrm{linv}(i)$ and $\mathrm{rinv}(i)$ for $i \in \mathrm{Mark}_n(C)$
- $\mathrm{coh}_{B,A}[f]$ for
    - Batanin tree B with $\dim B \leq n+1$
    - $A \in \mathrm{Full}_n(B)$
    - $f\colon \mathrm{Cptd}_{n+1}(\mathrm{Pos}(B)) \to C$

Cells and their boundaries

Definition

The set $\mathrm{Mark}_{n+1}(C)$ is inductively generated by:

- $\mathrm{igen}(u)$ for $u \in U^C_{n+1}$
- $\mathrm{lunit}(i)$ and $\mathrm{runit}(i)$ for $i \in \mathrm{Mark}_n(C)$

$$
\begin{aligned}
\mathrm{ibdry}\colon\ \mathrm{Mark}_{n+1}(C) &\to \mathrm{Sphere}_n(\mathrm{tr}_n\, C) \\
\mathrm{igen}(u) &\mapsto \phi^C_{n+1}(u) \\
\mathrm{lunit}(i) &\mapsto (\mathrm{linv}(i) \circ \mathrm{fwd}(i), 1) \\
\mathrm{runit}(i) &\mapsto (1, \mathrm{fwd}(i) \circ \mathrm{rinv}(i))
\end{aligned}
$$

Definition

The set $\mathrm{Cell}_{n+1}(C)$ is inductively generated by:

- $\mathrm{gen}(v)$ for $v \in V^C_{n+1}$
- $\mathrm{fwd}(i)$, $\mathrm{linv}(i)$ and $\mathrm{rinv}(i)$ for $i \in \mathrm{Mark}_n(C)$
- $\mathrm{coh}_{B,A}[f]$ for
  - Batanin tree B with $\dim B \leq n+1$
  - $A \in \mathrm{Full}_n(B)$
  - $f\colon\ \mathrm{Cptd}_{n+1}(\mathrm{Pos}(B)) \to C$

$$
\begin{aligned}
\mathrm{bdry}\colon\ \mathrm{Cell}_{n+1}(C) &\to \mathrm{Sphere}_n(\mathrm{tr}_n\, C) \\
\mathrm{gen}(v) &\mapsto \phi^C_{n+1}(v) \\
\mathrm{fwd}(i) &\mapsto \mathrm{ibdry}(i) \\
\mathrm{linv}(i) &\mapsto \mathrm{ibdry}(i)^{\mathrm{op}} \\
\mathrm{rinv}(i) &\mapsto \mathrm{ibdry}(i)^{\mathrm{op}} \\
\mathrm{coh}_{B,A}[f] &\mapsto \mathrm{Sphere}_n(\mathrm{tr}_n\, f)(A)
\end{aligned}
$$

Spheres and globular sets

Spheres are pairs of cells with the same source and target:

Definition

The set $\mathrm{Sphere}_{n+1}(C)$ and the projections natural transformations are defined by the following pullback square:

$$\begin{tikzcd}
\mathrm{Sphere}_{n+1}(C) \arrow[r, dashed, "\mathrm{pr}_2"] \arrow[d, dashed, "\mathrm{pr}_1"'] \arrow[dr, phantom, "\lrcorner", very near start] & \mathrm{Cell}_{n+1}(C) \arrow[d, "\mathrm{bdry}"] \\
\mathrm{Cell}_{n+1}(C) \arrow[r, "\mathrm{bdry}"'] & \mathrm{Sphere}_n(\mathrm{tr}_n\, C)
\end{tikzcd}$$

Spheres and globular sets

Spheres are pairs of cells with the same source and target:

Definition

The set $\mathsf{Sphere}_{n+1}(C)$ and the projections natural transformations are defined by the following pullback square:

$$\begin{tikzcd}
\mathsf{Sphere}_{n+1}(C) \arrow[r, dashed, "\mathsf{pr}_2"] \arrow[d, dashed, "\mathsf{pr}_1"'] \arrow[dr, phantom, "\lrcorner", very near start] & \mathsf{Cell}_{n+1}(C) \arrow[d, "\mathsf{bdry}"] \\
\mathsf{Cell}_{n+1}(C) \arrow[r, "\mathsf{bdry}"'] & \mathsf{Sphere}_n(\mathsf{tr}_n\, C)
\end{tikzcd}$$

Globular sets are the computads where the source and target of a generator are again generators:

Definition

The computad $\mathsf{Cptd}_{n+1}\, X$ for a globular set X consists of:

- the n-computad $\mathsf{Cptd}_n\, X$
- $V_{n+1} = X_{n+1}$
- $U_{n+1} = \emptyset$
- $\phi_{n+1}(x) = (\mathsf{gen}\,\mathsf{src}\,x, \mathsf{gen}\,\mathsf{tgt}\,x)$

Full spheres

Full spheres are an analogue of the admissible pairs of maps in the sense of Maltsiniotis [14]. They are the boundaries of the operations of ω-categories:

Definition

The support of a cell $c \in \mathrm{Cell}_n(C)$ is the set of generators that appear in c or its boundary. A sphere $A = (a, b)$ of $\mathrm{Cptd}_{n+1}\,\mathrm{Pos}(B)$ is full when

- the support of a contains all positions of $\partial_n^- B$
- the support of b contains all positions of $\partial_n^+ B$

Full spheres

Full spheres are an analogue of the admissible pairs of maps in the sense of Maltsiniotis [14]. They are the boundaries of the operations of ω-categories:

Definition

The support of a cell $c \in \mathrm{Cell}_n(C)$ is the set of generators that appear in c or its boundary. A sphere $A = (a, b)$ of $\mathrm{Cptd}_{n+1} \mathrm{Pos}(B)$ is full when

- the support of a contains all positions of $\partial_n^- B$
- the support of b contains all positions of $\partial_n^+ B$

$$B_1 = \overset{x}{\bullet} \xrightarrow{f} \overset{y}{\bullet} \xrightarrow{g} \overset{z}{\bullet}$$

$A_1 = (x, z)$ full

$A_2 = (z, x)$ not full

Full spheres

Full spheres are an analogue of the admissible pairs of maps in the sense of Maltsiniotis [14]. They are the boundaries of the operations of ω-categories:

Definition

The support of a cell $c \in \mathrm{Cell}_n(C)$ is the set of generators that appear in c or its boundary. A sphere $A = (a, b)$ of $\mathrm{Cptd}_{n+1} \mathrm{Pos}(B)$ is full when

- the support of a contains all positions of $\partial_n^- B$
- the support of b contains all positions of $\partial_n^+ B$

$$B_1 = \overset{x}{\bullet} \xrightarrow{f} \overset{y}{\bullet} \xrightarrow{g} \overset{z}{\bullet}$$

$$A_1 = (x, z) \quad \text{full}$$

$$A_2 = (z, x) \quad \text{not full}$$

$$B_2 = \overset{x}{\bullet} \xrightarrow{f} \overset{y}{\bullet} \xrightarrow{g} \overset{z}{\bullet} \xrightarrow{h} \overset{w}{\bullet}$$

$$A_3 = (x, w) \quad \text{full}$$

Full spheres

Full spheres are an analogue of the admissible pairs of maps in the sense of Maltsiniotis [14]. They are the boundaries of the operations of ω-categories:

Definition

The support of a cell $c \in \mathrm{Cell}_n(C)$ is the set of generators that appear in c or its boundary. A sphere $A = (a, b)$ of $\mathrm{Cptd}_{n+1} \operatorname{Pos}(B)$ is full when

- the support of a contains all positions of $\partial_n^- B$
- the support of b contains all positions of $\partial_n^+ B$

$$B_1 = \overset{x}{\bullet} \xrightarrow{f} \overset{y}{\bullet} \xrightarrow{g} \overset{z}{\bullet}$$

$$A_1 = (x, z) \quad \text{full}$$

$$A_2 = (z, x) \quad \text{not full}$$

$$B_2 = \overset{x}{\bullet} \xrightarrow{f} \overset{y}{\bullet} \xrightarrow{g} \overset{z}{\bullet} \xrightarrow{h} \overset{w}{\bullet}$$

$$A_3 = (x, w) \quad \text{full}$$

$$A_4 = (f \circ g \circ h, (f \circ g) \circ h) \quad \text{full}$$

$$A_5 = ((f \circ g) \circ h, f \circ g \circ h) \quad \text{full}$$

$$\begin{aligned} f \circ g &= \mathrm{coh}_{B_1, A_1}[f \mapsto f, g \mapsto g] \\ (f \circ g) \circ h &= \mathrm{coh}_{B_1, A_1}[f \mapsto f \circ g, g \mapsto h] \\ f \circ g \circ h &= \mathrm{coh}_{B_2, A_3}[\mathrm{id}] \end{aligned}$$

Computads and ω-categories

Definition

The category Comp of computads with invertible generators is the limit:

$$\dots \xrightarrow{\mathrm{tr}_n} \mathrm{Comp}_n \xrightarrow{\mathrm{tr}_{n-1}} \dots \xrightarrow{\mathrm{tr}_1} \mathrm{Comp}_1 \xrightarrow{\mathrm{tr}_0} \mathrm{Comp}_0$$

Computads and ω-categories

Definition

The category Comp of computads with invertible generators is the limit:

$$\dots \xrightarrow{\mathrm{tr}_n} \mathrm{Comp}_n \xrightarrow{\mathrm{tr}_{n-1}} \dots \xrightarrow{\mathrm{tr}_1} \mathrm{Comp}_1 \xrightarrow{\mathrm{tr}_0} \mathrm{Comp}_0$$

The remaining data gives rise to an adjunction:

$$\begin{tikzcd} \mathrm{Glob} \arrow[r, shift left=2, "\mathrm{Cptd}"] \arrow[r, phantom, "\perp"] & \mathrm{Comp} \arrow[l, shift left=2, "\mathrm{Cell}"] \end{tikzcd}$$

and hence a monad T on globular sets.

Computads and ω-categories

Definition

The category Comp of computads with invertible generators is the limit:

$$\dots \xrightarrow{\mathrm{tr}_n} \mathrm{Comp}_n \xrightarrow{\mathrm{tr}_{n-1}} \dots \xrightarrow{\mathrm{tr}_1} \mathrm{Comp}_1 \xrightarrow{\mathrm{tr}_0} \mathrm{Comp}_0$$

The remaining data gives rise to an adjunction:

$$\begin{tikzcd} \mathrm{Glob} \arrow[r, shift left, "\mathrm{Cptd}"] \arrow[r, phantom, "\perp"] & \mathrm{Comp} \arrow[l, shift left, "\mathrm{Cell}"] \end{tikzcd}$$

and hence a monad T on globular sets.

Theorem

The category of weak ω-categories is the category of T-algebras.

$$\begin{tikzcd} \mathrm{Comp} \arrow[rr, dashed, "K"] \arrow[dr, shift left, "\mathrm{Cell}"] & & \mathrm{Cat}_\omega \arrow[dl, shift left, "U^T"] \\ & \mathrm{Glob} \arrow[ul, shift left, "\mathrm{Cptd}"] \arrow[ur, shift left, "F^T"] & \end{tikzcd}$$

Variants of computads

Definition

An ordinary computad is a computad with no invertible generators.

Definition

A rigid morphism $f \colon C \to D$ is one that sends generators to generators.

Variants of computads

Definition

An ordinary computad is a computad with no invertible generators.

Definition

A rigid morphism $f \colon C \to D$ is one that sends generators to generators.

$$\begin{tikzcd}
\mathrm{Comp}^{\mathrm{ord},\mathrm{rig}} \arrow[r] \arrow[d, hook] \arrow[dr, phantom, "\lrcorner", very near start] & \mathrm{Comp}^{\mathrm{ord}} \arrow[d, hook] \\
\mathrm{Comp}^{\mathrm{rig}} \arrow[r] & \mathrm{Comp}
\end{tikzcd}$$

Variants of computads

Definition

An ordinary computad is a computad with no invertible generators.

Definition

A rigid morphism $f\colon C \to D$ is one that sends generators to generators.

$$\begin{tikzcd} \mathrm{Comp}^{\mathrm{ord,rig}} \arrow[r] \arrow[d, hook] \arrow[dr, phantom, very near start, "\lrcorner"] & \mathrm{Comp}^{\mathrm{ord}} \arrow[d, hook] \\ \mathrm{Comp}^{\mathrm{rig}} \arrow[r] & \mathrm{Comp} \end{tikzcd}$$

Theorem

The rigid subcategories are presheaf topoi.

Weak equivalences and trivial fibrations

Definition

A weak equivalence $f\colon X \to Y$ between ω-categories is a morphism satisfying that:

- for every $y \in Y_0$ there exists $x \in X_0$ and and $f(x) \xrightarrow{\sim} y$
- for every $x, x' \in X_0$ and $y\colon f(x) \to f(x')$, there exists $g\colon x \to x'$ and $f(g) \xrightarrow{\sim} y$

$$\begin{tikzcd}
x \arrow[r, dashed, "g"] \arrow[d, mapsto] & x' \arrow[d, mapsto] \\
f(x) \arrow[r, dashed, bend left, "f(g)"{name=U}] \arrow[r, bend right, "y"'{name=D}] & f(x')
\arrow[from=U, to=D, Rightarrow, dashed, "\wr"]
\end{tikzcd}$$

$$\begin{tikzcd}
F^T\mathbb{S}^n \arrow[r, "{(x,x')}"] \arrow[d, hook] & X \arrow[d, "f"] \\
F^T\mathbb{D}^{n+1} \arrow[r, "y"'] \arrow[ur, dashed, "g", "\sim"'] & Y
\end{tikzcd}$$

Definition

A trivial fibration is a morphism that has the right lifting property with respect to $F^T\mathbb{S}^n \hookrightarrow F^T\mathbb{D}^{n+1}$ for all $n \geq -1$.

Every ω-category is free

Theorem

Every ω-category is weakly equivalent to one generated by a computad.

Idea of proof.

- The inclusion $\mathrm{Comp}^{\mathrm{ord,rig}} \hookrightarrow \mathrm{Comp} \xrightarrow{K} \mathrm{Cat}_\omega$ admits a right adjoint Q.
- The counit $KQX \to X$ is a trivial fibration and hence a weak equivalence. $\square$

Every ω-category is free

Theorem

Every ω-category is weakly equivalent to one generated by a computad.

Idea of proof.

- The inclusion $\mathrm{Comp}^{\mathrm{ord,rig}} \hookrightarrow \mathrm{Comp} \xrightarrow{K} \mathrm{Cat}_\omega$ admits a right adjoint Q.
- The counit $KQX \to X$ is a trivial fibration and hence a weak equivalence. $\square$

Remark

Computads are the cofibrant objects for the factorisation system generated by the sphere to disk inclusions.

Did we need invertibility?

Proposition

The inclusion $\mathsf{Comp}^{\mathsf{ord}} \hookrightarrow \mathsf{Comp}$ admits a right adjoint R that preserves the generated ω-category.

Idea of proof.

We construct the computad RC by induction on the dimension. Each element of $\mathsf{Mark}_n(C)$ gives rise to three generators of RC, one for its itself and two for its inverses. □

Nonetheless, we get a very succinct description of certain weak ω-categories by allowing invertible generators. For example, we can define the walking isomorphism $\mathbb{E}^{n+1}$ to be the computad extending $\mathbb{S}^n$ with a single invertible generator.

Future directions and consequences

- This work is part of our pursuit of a model structure on Cat_ω.

Future directions and consequences

- This work is part of our pursuit of a model structure on Cat_ω.
- What remaining to be done is showing that certain cylinder objects exist.

Future directions and consequences

- This work is part of our pursuit of a model structure on Cat_ω.
- What remaining to be done is showing that certain cylinder objects exist.
- This is a stepping stone towards comparing (marked) ω-categories with other models of (∞, ∞)-categories, such as Θ-spaces, or complicial sets.

Future directions and consequences

- This work is part of our pursuit of a model structure on Cat_ω.
- What remaining to be done is showing that certain cylinder objects exist.
- This is a stepping stone towards comparing (marked) ω-categories with other models of (∞, ∞)-categories, such as Θ-spaces, or complicial sets.
- Extending the theory to ω-groupoids would yield a new model for homotopy types, giving new insight on their algebraic structure and strengthening their connection to intensional type theory.

Thank you for your attention!

References i

[1] Dimitri Ara. “Sur les ∞-groupoïdes de Grothendieck et une variante ∞-catégorique”. Thèse de doctorat. 2010.

[2] Michael A. Batanin. *Computads and slices of operads*. 2002. Pre-published.

[3] Michael A. Batanin. “Monoidal Globular Categories As a Natural Environment for the Theory of Weak n-Categories”. In: *Advances in Mathematics* 136.1 (1998).

[4] Thibaut Benjamin, Camil Champin, and Ioannis M. *Computads with Invertible Generators for Weak ω-Categories*. 2026. Pre-published.

[5] Thibaut Benjamin, Ioannis M., and Chiara Sarti. “CaTT contexts are finite computads”. In: *Proceedings of the 40th Conference on Mathematical Foundations of Programming Semantics*. 2024.

[6] Thibaut Benjamin, Samuel Mimram, and Eric Finster. “Globular weak ω-categories as models of a type theory”. In: *Higher Structures* 2 (2024), pp. 1–69.

[7] John Bourke. “Iterated algebraic injectivity and the faithfulness conjecture”. In: *Higher Structures* 4.2 (2020).

[8] Albert Burroni. “Higher-dimensional word problems with applications to equational logic”. In: *Theoretical Computer Science* 115.1 (1993), pp. 43–62.

[9] Christopher J. Dean et al. “Computads for weak ω-categories as an inductive type”. In: *Advances in Mathematics* 450 (2024).

References ii

[10] Eric Finster and Samuel Mimram. "A type-theoretical definition of weak ω-categories". In: *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science*. 2017.

[11] Alexander Grothendieck. *Pursuing Stacks*. 1983.

[12] Yves Lafont, François Métayer, and Krzysztof Worytkiewicz. "A folk model structure on omega-cat". In: *Advances in Mathematics* 224.3 (2010).

[13] Tom Leinster. *Higher operads, higher categories*. 298. Cambridge University Press, 2004.

[14] Georges Maltsiniotis. *Grothendieck ∞-groupoids, and still another definition of ∞-categories*. 2010.

[15] Ross Street. "Limits indexed by category-valued 2-functors". In: *Journal of Pure and Applied Algebra* 8.2 (1976).

Bonus: The grammar of CaTT

We can alternatively reason about weak ω-categories and coinductive invertibilty using dependent type theory. This allows us to construct proof assistants for reasoning about weak ω-categories.

Bonus: The grammar of CaTT

We can alternatively reason about weak ω-categories and coinductive invertibilty using dependent type theory. This allows us to construct proof assistants for reasoning about weak ω-categories. The grammar of the type theory CaTT:

$$
\begin{array}{lrcccc}
\text{Variables} & x, y & ::= & x \in \mathbb{N} & & \\
\text{Terms} & t, u & ::= & x & \mid & \mathsf{coh}_{\Delta,A}[\sigma] \\
\text{Types} & A, B & ::= & \star & \mid & u \to_A v. \\
\text{Contexts} & \Gamma, \Delta & ::= & \emptyset & \mid & \Gamma, x : A \\
\text{Substitutions} & \sigma, \tau & ::= & \langle\rangle & \mid & \langle \sigma, x \mapsto t \rangle
\end{array}
$$

Bonus: The grammar of CaTT

We can alternatively reason about weak ω-categories and coinductive invertibilty using dependent type theory. This allows us to construct proof assistants for reasoning about weak ω-categories. The grammar of the type theory CaTT:

$$
\begin{array}{lrcccc}
\text{Variables} & x, y & ::= & x \in \mathbb{N} & & \\
\text{Terms} & t, u & ::= & x & \mid & \mathsf{coh}_{\Delta, A}[\sigma] \\
\text{Types} & A, B & ::= & \star & \mid & u \to_A v. \\
\text{Contexts} & \Gamma, \Delta & ::= & \emptyset & \mid & \Gamma, x \colon A \\
\text{Substitutions} & \sigma, \tau & ::= & \langle\rangle & \mid & \langle \sigma, x \mapsto t \rangle
\end{array}
$$

Its judgement forms are the usual ones for dependent type theory:

$$\Gamma \vdash \qquad\qquad \Gamma \vdash A \qquad\qquad \Gamma \vdash t \colon A \qquad\qquad \Gamma \vdash \sigma \colon \Delta$$

with three additional auxiliary judgement forms:

$$\Gamma \vdash_{\mathsf{ps}} \qquad\qquad \Gamma \vdash_{\mathsf{ps}} x \colon A \qquad\qquad \Gamma \vdash_{\mathsf{full}} A$$

The rules of CaTT

The rules of CaTT contain the usual structural rules for contexts, substitutions and variables. Its type and term formation rules are the following ones:

$$\frac{\Gamma \vdash}{\Gamma \vdash \star} \qquad \frac{\Gamma \vdash u : A \qquad \Gamma \vdash v : A}{\Gamma \vdash u \to_A v} \qquad \frac{\Gamma \vdash_{\mathsf{ps}} \qquad \Gamma \vdash_{\mathsf{full}} A \qquad \Delta \vdash \delta : \Gamma}{\Delta \vdash \mathsf{coh}_{\Gamma,A}[\sigma] : A[\sigma]}$$

The rules for pasting contexts and full types are more involved, but analogous to the ones described in the setting of computads.

Theorem

Contexts of CaTT are finite computads and models of CaTT are weak ω-categories.

An extension of CaTT

We may extend CaTT with a new type former of invertible terms and six destructors:

$$\frac{\Gamma \vdash f \colon x \to_A y}{\Gamma \vdash \mathsf{Inv}(f)} \qquad \frac{\Gamma \vdash i : \mathsf{Inv}_A(f)}{\begin{array}{rr} \Gamma \vdash \mathsf{linv}(i) : A^{\mathsf{op}} & \Gamma \vdash \mathsf{rinv}(i) : A^{\mathsf{op}} \\ \Gamma \vdash \mathsf{lunit}(i) : \mathsf{linv}(i) * f \to \mathsf{id} & \Gamma \vdash \mathsf{runit}(i) : \mathsf{id} \to f * \mathsf{rinv}(i) \\ \Gamma \vdash \mathsf{ilunit}(i) : \mathsf{Inv}(\mathsf{lunit}(i)) & \Gamma \vdash \mathsf{irunit}(i) : \mathsf{Inv}(\mathsf{runit}(i)) \end{array}}$$

We also add a constructor and a recursion principle as well as β and η rules.

Semantics

$$\begin{tikzcd}
S_{CaTT} \arrow[r, hook] \arrow[d, hook] & S_{ICaTT} \arrow[d, hook] \\
\mathrm{Mod}(CaTT) \arrow[r, shift left] \arrow[r, phantom, "\perp"] & \mathrm{Mod}(ICaTT) \arrow[l, shift left]
\end{tikzcd}$$

Semantics

$$\begin{tikzcd} S_{CaTT} \arrow[r, hook] \arrow[d, hook] & S_{ICaTT} \arrow[d, hook] \\ \mathrm{Mod}(CaTT) \arrow[r, shift left] \arrow[r, phantom, "\perp"] & \mathrm{Mod}(ICaTT) \arrow[l, shift left] \end{tikzcd}$$

Theorem

The extension of CaTT with invertible terms is conservative.

Semantics

$$\begin{tikzcd} S_{CaTT} \arrow[r, hook] \arrow[d, hook] & S_{ICaTT} \arrow[d, hook] \\ \mathrm{Mod}(CaTT) \arrow[r, shift left] \arrow[r, phantom, "\perp"] & \mathrm{Mod}(ICaTT) \arrow[l, hook, shift left] \end{tikzcd}$$

Theorem

The extension of CaTT with invertible terms is conservative.

Remark

Via the adjunction on the left we can describe some infinite computads as contexts of the extended type theory.

Semantics

$$\begin{tikzcd} S_{CaTT} \arrow[r, hook] \arrow[d, hook] & S_{ICaTT} \arrow[d, hook] \\ \mathrm{Mod}(CaTT) \arrow[r, shift left] \arrow[r, phantom, "\perp"] & \mathrm{Mod}(ICaTT) \arrow[l, shift left, hook] \end{tikzcd}$$

Theorem

The extension of CaTT with invertible terms is conservative.

Remark

Via the adjunction on the left we can describe some infinite computads as contexts of the extended type theory.

Theorem

We have semantics in fibrant marked weak ω-categories.