Strictly associative and unital higher category theory

Jamie Vicary
School of Computer Science, University of Birmingham

(With Christoph Dorn, Christopher Douglas, and David Reutter)

Midland Graduate School Christmas Seminar
University of Birmingham, UK
19 December 2018
Working with higher structures

Higher structures are important for several areas of mathematics and computer science, including homotopy type theory, quantum field theory, manifold theory, and representation theory.
Working with higher structures

Higher structures are important for several areas of mathematics and computer science, including homotopy type theory, quantum field theory, manifold theory, and representation theory. However, working with these structures poses many difficulties.
Working with higher structures

Higher structures are important for several areas of mathematics and computer science, including homotopy type theory, quantum field theory, manifold theory, and representation theory.

However, working with these structures poses many difficulties.

▶ How can we formally define the structure we are working with (higher proof, manifold, 2-group, knot, etc)?
Working with higher structures

Higher structures are important for several areas of mathematics and computer science, including homotopy type theory, quantum field theory, manifold theory, and representation theory.

However, working with these structures poses many difficulties.

- How can we formally define the structure we are working with (higher proof, manifold, 2-group, knot, etc)?
- How can we communicate it to collaborators and readers, and learn more about it ourselves?
Working with higher structures

Higher structures are important for several areas of mathematics and computer science, including homotopy type theory, quantum field theory, manifold theory, and representation theory.

However, working with these structures poses many difficulties.

▶ How can we formally define the structure we are working with (higher proof, manifold, 2-group, knot, etc)?
▶ How can we communicate it to collaborators and readers, and learn more about it ourselves?
▶ How can we modify it, discover its properties, and prove theorems about it?
Higher structures are important for several areas of mathematics and computer science, including homotopy type theory, quantum field theory, manifold theory, and representation theory.

However, working with these structures poses many difficulties.

▶ How can we formally define the structure we are working with (higher proof, manifold, 2-group, knot, etc)?
▶ How can we communicate it to collaborators and readers, and learn more about it ourselves?
▶ How can we modify it, discover its properties, and prove theorems about it?
▶ How can we use computers to help us achieve these goals, not only theoretically, but practically?
Proof assistants for higher algebra

Several proof assistants for higher category theory exist.
Proof assistants for higher algebra

Several proof assistants for higher category theory exist.

- Homotopy type theory in Agda, Coq (2010)
Several proof assistants for higher category theory exist.

- Homotopy type theory in Agda, Coq (2010)
- Opetopic by Eric Finster (2013)
Proof assistants for higher algebra

Several proof assistants for higher category theory exist.

▶ Homotopy type theory in Agda, Coq (2010)

▶ Opetopic by Eric Finster (2013)

▶ Finster-Mimram contractible $\infty$-categories (2017)
Proof assistants for higher algebra

Several proof assistants for higher category theory exist.

- Homotopy type theory in Agda, Coq (2010)
- Opetopic by Eric Finster (2013)
- Finster-Mimram contractible $\infty$-categories (2017)
- Globular by JV and collaborators (2015)
Proof assistants for higher algebra

Several proof assistants for higher category theory exist.

- Homotopy type theory in Agda, Coq (2010)
- *Opetopic* by Eric Finster (2013)
- Finster-Mimram contractible $\infty$-categories (2017)
- *Globular* by JV and collaborators (2015)

Our goal: easy construction, visualization and manipulation of complex proofs in arbitrary dimension. None of these achieve this.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

\[
\begin{array}{c}
A \\
\downarrow \ s \\
B \\
\uparrow \ C \\
D \\
\downarrow \ t \\
E \\
\end{array} 
\]
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

- Idea of formal graphical calculus go back to Penrose in 1971.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

- Idea of formal graphical calculus go back to Penrose in 1971.
- Strict associativity and unitality comes built in.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

▶ Idea of formal graphical calculus go back to Penrose in 1971.
▶ Strict associativity and unitality comes built in.
▶ Joyal and Street developed these ideas for monoidal categories with braiding and symmetry in 1991.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

- Idea of formal graphical calculus go back to Penrose in 1971.
- Strict associativity and unitality comes built in.
- Joyal and Street developed these ideas for monoidal categories with braiding and symmetry in 1991.
- Extension to Gray categories (special case of $n = 3$) given sketch proof by Barrett, Meusburger and Schaumann in 2012.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

- Idea of formal graphical calculus go back to Penrose in 1971.
- Strict associativity and unitality comes built in.
- Joyal and Street developed these ideas for monoidal categories with braiding and symmetry in 1991.
- Extension to Gray categories (special case of $n = 3$) given sketch proof by Barrett, Meusburger and Schaumann in 2012.
- In higher dimensions, no formal theory has been developed.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

- Idea of formal graphical calculus go back to Penrose in 1971.
- Strict associativity and unitality comes built in.
- Joyal and Street developed these ideas for monoidal categories with braiding and symmetry in 1991.
- Extension to Gray categories (special case of $n = 3$) given sketch proof by Barrett, Meusburger and Schaumann in 2012.
- In higher dimensions, no formal theory has been developed.
- Nonetheless, regularly used as an informal language.
Graphical calculus

It is conjectured that $n$-categories have an $n$-dimensional graphical calculus, which is the dual of the ordinary ‘commutative diagrams’.

- Idea of formal graphical calculus go back to Penrose in 1971.
- Strict associativity and unitality comes built in.
- Joyal and Street developed these ideas for monoidal categories with braiding and symmetry in 1991.
- Extension to Gray categories (special case of $n=3$) given sketch proof by Barrett, Meusburger and Schaumann in 2012.
- In higher dimensions, no formal theory has been developed.
- Nonetheless, regularly used as an informal language.

Scott Carter, “Turning a sphere from red to blue”
Homotopy

The graphical calculus suggests homotopy as a basic mechanism with which to manipulate terms and execute computations.
Homotopy

The graphical calculus suggests homotopy as a basic mechanism with which to manipulate terms and execute computations. The ‘strictest’ definitions of higher category do not allow these manipulations, and are known to be insufficiently general.
Homotopy

The graphical calculus suggests *homotopy* as a basic mechanism with which to manipulate terms and execute computations.

The ‘strictest’ definitions of higher category do not allow these manipulations, and are known to be insufficiently general.

At the opposite end of the spectrum, the ‘weakest’ definitions allow not only these manipulations, but far more besides.
Homotopy

The graphical calculus suggests homotopy as a basic mechanism with which to manipulate terms and execute computations.

The ‘strictest’ definitions of higher category do not allow these manipulations, and are known to be insufficiently general.

At the opposite end of the spectrum, the ‘weakest’ definitions allow not only these manipulations, but far more besides.

A model of higher categories is semistrict if it is as strict as possible, while still allowing arbitrary homotopy. However, yields long proofs.
Homotopy

The graphical calculus suggests homotopy as a basic mechanism with which to manipulate terms and execute computations.

The ‘strictest’ definitions of higher category do not allow these manipulations, and are known to be insufficiently general.

At the opposite end of the spectrum, the ‘weakest’ definitions allow not only these manipulations, but far more besides.

<table>
<thead>
<tr>
<th>Weak</th>
<th>Semiweak</th>
<th>Semistrict</th>
<th>Strict</th>
</tr>
</thead>
</table>

A model of higher categories is semistrict if it is as strict as possible, while still allowing arbitrary homotopy. However, yields long proofs.

A definition is semiweak if it is as weak as possible, while still being strictly associative and unital; i.e. composites are unique.
Associative $n$-categories

We propose a new semiweak approach to higher category theory called associative $n$-categories.
Associative $n$-categories

We propose a new semiweak approach to higher category theory called associative $n$-categories.

- Simple combinatorial model gives an elementary way to encode composites in a free $(\infty, \infty)$-category.
We propose a new semiweak approach to higher category theory called **associative n-categories**.

- Simple combinatorial model gives an elementary way to encode composites in a free \((\infty, \infty)\)-category.

- Immediate string diagram representation.
We propose a new semiweak approach to higher category theory called *associative* $n$-*categories*.

- Simple combinatorial model gives an elementary way to encode composites in a free $(\infty, \infty)$-*category*.
- Immediate string diagram representation.
- Composition is strictly associative and unital.
We propose a new semiweak approach to higher category theory called associative $n$-categories.

- Simple combinatorial model gives an elementary way to encode composites in a free $(\infty, \infty)$-category.
- Immediate string diagram representation.
- Composition is strictly associative and unital.
- All the weak structure is in homotopies of composites.
Associative $n$-categories

We propose a new semiweak approach to higher category theory called associative $n$-categories.

- Simple combinatorial model gives an elementary way to encode composites in a free $(\infty, \infty)$-category.
- Immediate string diagram representation.
- Composition is strictly associative and unital.
- All the weak structure is in homotopies of composites.
- Amenable to computer implementation.
We propose a new semiweak approach to higher category theory called *associative n-categories*.

- Simple combinatorial model gives an elementary way to encode composites in a free \((\infty, \infty)\)-category.
- Immediate string diagram representation.
- Composition is strictly associative and unital.
- All the weak structure is in homotopies of composites.
- Amenable to computer implementation.
- High-level methods allow easy homotopy construction.
Monotone functions

Consider the following string diagram in a bicategory.
Monotone functions

Consider the following string diagram in a bicategory. It gives rise to a sequence of cospans of monotone functions.
Definition. Given a category $\mathcal{C}$, with $A, B \in \text{Ob}(\mathcal{C})$, we define the category of iterated cospans $\text{IC}_{\mathcal{C}}(A, B)$ as follows:
Diagrams from iterated cospans

Definition. Given a category $\mathcal{C}$, with $A, B \in \text{Ob}(\mathcal{C})$, we define the category of iterated cospans $\text{IC}_\mathcal{C}(A, B)$ as follows:

- an object is a sequence of cospans in $\mathcal{C}$, from $A$ to $B$;

\[
A \rightarrow P_0 \leftarrow V_1 \rightarrow P_1 \leftarrow V_2 \rightarrow P_2 \leftarrow V_3 \rightarrow P_3 \leftarrow B
\]
Definition. Given a category $\mathcal{C}$, with $A, B \in \text{Ob}(\mathcal{C})$, we define the category of iterated cospans $\mathcal{IC}_\mathcal{C}(A, B)$ as follows:

- an object is a sequence of cospans in $\mathcal{C}$, from $A$ to $B$;
- a morphism is:

\[
\begin{align*}
A & \rightarrow P_0 \leftarrow V_1 \rightarrow P_1 \leftarrow V_2 \rightarrow P_2 \leftarrow V_3 \rightarrow P_3 \leftarrow B \\
A' & \rightarrow P'_0 \leftarrow V'_1 \rightarrow P'_1 \leftarrow V_2 \rightarrow P'_2 \leftarrow B'
\end{align*}
\]
Definition. Given a category $\mathcal{C}$, with $A, B \in \text{Ob}(\mathcal{C})$, we define the category of iterated cospans $\text{IC}_\mathcal{C}(A, B)$ as follows:

- an object is a sequence of cospans in $\mathcal{C}$, from $A$ to $B$;
- a morphism is:
  - a monotone function between peaks,
Definition. Given a category $\mathcal{C}$, with $A, B \in \text{Ob}(\mathcal{C})$, we define the category of iterated cospans $\mathcal{IC}_\mathcal{C}(A, B)$ as follows:

- an object is a sequence of cospans in $\mathcal{C}$, from $A$ to $B$;
- a morphism is:
  - a monotone function between peaks,
  - built from morphisms of $\mathcal{C}$,
Diagrams from iterated cospans

**Definition.** Given a category $C$, with $A, B \in \text{Ob}(C)$, we define the category of iterated cospans $IC_C(A, B)$ as follows:

- an object is a sequence of cospans in $C$, from $A$ to $B$;
- a morphism is:
  - a monotone function between peaks,
  - built from morphisms of $C$,
  - with identities between valleys,
Diagrams from iterated cospans

**Definition.** Given a category $C$, with $A, B \in \text{Ob}(C)$, we define the category of iterated cospans $IC_C(A, B)$ as follows:

- an object is a sequence of cospans in $C$, from $A$ to $B$;
- a morphism is:
  - a monotone function between peaks,
  - built from morphisms of $C$,
  - with identities between valleys,
  - such that all squares commute.
Definition. Given a category $C$, with $A, B \in \text{Ob}(C)$, we define the category of iterated cospans $IC_C(A, B)$ as follows:

- an object is a sequence of cospans in $C$, from $A$ to $B$;
- a morphism is:
  - a monotone function between peaks,
  - built from morphisms of $C$,
  - with identities between valleys,
  - such that all squares commute.

\[
\begin{align*}
A & \to P_0 \leftarrow V_1 \to P_1 \leftarrow V_2 \to P_2 \leftarrow V_3 \to P_3 \leftarrow B \\
& \downarrow \sqrt{} \quad \downarrow \sqrt{} \quad \downarrow \sqrt{} \quad \downarrow \sqrt{} \quad \downarrow \sqrt{} \quad \downarrow \sqrt{} \quad \downarrow \sqrt{} \quad \downarrow \sqrt{} \quad \downarrow \sqrt{} \\
A' & \to P_0' \leftarrow V_1' \to P_1' \leftarrow V_2 \to P_2 \leftarrow B'
\end{align*}
\]

Definition. An $n$-diagram is an object of some category obtained by starting with 1, and applying the IC construction $n$ times.
Diagrams from iterated cospans

Here are 53 examples of 0-diagrams, all objects of 1:
Diagrams from iterated cospans

Here are 7 examples of 1-diagrams, all objects of $IC_1(\bullet, \bullet) = \Delta$: 

\begin{align*}
\bullet & \longrightarrow \bullet & & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet & \longrightarrow & \bullet & \longleftarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet \\
\bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet
\end{align*}
Diagrams from iterated cospans

Here is 1 example of a 2-diagram, an object of $IC_\Delta(3, 3)$:
Types

These diagrams are *untyped*.
Types

These diagrams are *untyped*. To add types, we decorate with labels.
Types

These diagrams are *untyped*. To add types, we decorate with labels. The standard graphical calculus is just a prettier version of this.
Types

These diagrams are *untyped*. To add types, we decorate with labels. The standard graphical calculus is just a prettier version of this.

*Type checking* verifies that label neighbourhoods match definition.
Types

These diagrams are *untyped*. To add types, we decorate with labels. The standard graphical calculus is just a prettier version of this.

*Type checking* verifies that label neighbourhoods match definition.
Types

These diagrams are *untyped*. To add types, we decorate with labels. The standard graphical calculus is just a prettier version of this.

Type checking verifies that label neighbourhoods match definition.
Homotopies

The following homotopy is an interchanger:
Homotopies

The following homotopy is an interchanger:

It exists as a morphism in $IC_\Delta(2, 2)$. 
Homotopies

The following homotopy is an interchanger:

\[
\begin{array}{cccc}
B & f & L & f & R \\
\downarrow & & \downarrow & & \downarrow \\
B & f & L & \tau & R \\
\downarrow & & \downarrow & & \downarrow \\
B & f & L & f & R \\
\downarrow & & \downarrow & & \downarrow \\
B & \sigma & L & f & R \\
\downarrow & & \downarrow & & \downarrow \\
B & f & L & f & R \\
\end{array}
\]

It exists as a morphism in \( IC_{\Delta}(2, 2) \).

It is well-typed:
Homotopies

The following homotopy is an interchanger:

It exists as a morphism in $IC_\Delta(2, 2)$.

It is well-typed:
Homotopies

The following homotopy is an interchanger:

It exists as a morphism in $IC_\Delta(2, 2)$.

It is well-typed:

- The neighbourhood of $\sigma$ normalizes to its standard form.
Homotopies

The following homotopy is an interchanger:

It exists as a morphism in $IC_\Delta(2, 2)$.

It is well-typed:

- The neighbourhood of $\sigma$ normalizes to its standard form.
- The neighbourhood of $\tau$ normalizes to its standard form.
Homotopies

The following homotopy is an interchanger:

It exists as a morphism in $IC_\Delta(2, 2)$.

It is well-typed:

- The neighbourhood of $\sigma$ normalizes to its standard form.
- The neighbourhood of $\tau$ normalizes to its standard form.

So homotopies are built in to associative $n$-categories.
High-level methods

This low-level definition is too unwieldy for direct manipulation.
High-level methods

This low-level definition is too unwieldy for direct manipulation. 
*High-level methods* are needed to work with these structures.
High-level methods

This low-level definition is too unwieldy for direct manipulation. *High-level methods* are needed to work with these structures. Consider the problem of contracting this composite:
High-level methods

This low-level definition is too unwieldy for direct manipulation. *High-level methods* are needed to work with these structures.

Consider the problem of contracting this composite:
High-level methods

This low-level definition is too unwieldy for direct manipulation. *High-level methods* are needed to work with these structures.

Consider the problem of contracting this composite:
High-level methods

This low-level definition is too unwieldy for direct manipulation. *High-level methods* are needed to work with these structures.

Consider the problem of contracting this composite:

We build the contraction as a pushout of cospans.
High-level methods

This low-level definition is too unwieldy for direct manipulation. *High-level methods* are needed to work with these structures.

Consider the problem of contracting this composite:

We build the contraction as a pushout of cospans.
High-level methods

This low-level definition is too unwieldy for direct manipulation. *High-level methods* are needed to work with these structures.

Consider the problem of contracting this composite:

We build the contraction as a pushout of cospans.

Gives an insight into the relationship with virtual $n$-categories.
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in *homotopy.io*, along with high-level methods allowing efficient homotopy construction in all dimensions.
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.

- Implemented in *homotopy.io*, along with high-level methods allowing efficient homotopy construction in all dimensions.

- Many exciting things on our theory roadmap:
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in *homotopy.io*, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in homotopy.io, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in homotopy.io, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in *homotopy.io*, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
  - Tactics and automation
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in homotopy.io, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
  - Tactics and automation
- There will also be a lot of activity on the ‘soft’ side:
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in homotopy.io, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
  - Tactics and automation
- There will also be a lot of activity on the ‘soft’ side:
  - Animation of proofs, export to YouTube
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in *homotopy.io*, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
  - Tactics and automation
- There will also be a lot of activity on the ‘soft’ side:
  - Animation of proofs, export to YouTube
  - Virtual reality, immersive interaction with proofs in 3d
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in *homotopy.io*, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
  - Tactics and automation
- There will also be a lot of activity on the ‘soft’ side:
  - Animation of proofs, export to YouTube
  - Virtual reality, immersive interaction with proofs in 3d
  - 3d printing of geometries
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in *homotopy.io*, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
  - Tactics and automation
- There will also be a lot of activity on the ‘soft’ side:
  - Animation of proofs, export to YouTube
  - Virtual reality, immersive interaction with proofs in 3d
  - 3d printing of geometries
- Lots of opportunity for collaboration.
Outlook

- Associative $n$-categories give a new definition of free $(\infty, \infty)$-categories which are strictly associative and unital.
- Implemented in homotopy.io, along with high-level methods allowing efficient homotopy construction in all dimensions.
- Many exciting things on our theory roadmap:
  - Going from globular structure to cubical structure
  - Dualizability and invertibility via fibrational typing
  - Logical structure (limits and colimits, dependent types)
  - Tactics and automation
- There will also be a lot of activity on the ‘soft’ side:
  - Animation of proofs, export to YouTube
  - Virtual reality, immersive interaction with proofs in 3d
  - 3d printing of geometries
- Lots of opportunity for collaboration.

Thank you!