Coherence for braided and symmetric pseudomonoids

Dominic Verdon

Department of Computer Science
University of Oxford

17/09/2017 / PSSL2017
Outline

1. Background
   - Result and motivation
     - Full generality from semistrictness
     - Introduction to semistrict bicategories: Gray monoids
     - Quasistrict braiding and symmetry

2. Coherence for pseudomonoids
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitors
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
Coherence theorems for braided and symmetric pseudomonoids

- Braided/symmetric *pseudomonoids* are categorifications of commutative monoids (e.g. braided/symmetric monoidal categories in **Cat**, braided/symmetric 2-vector space, etc).
- We prove coherence theorems (biequivalences) for braided and symmetric pseudomonoids.
- Generalises MacLane’s theorems [4] — in fact, proves them with string diagrams.
- Categorifies PROs, PROBs, PROPs for monoids and commutative monoids.
Motivation: higher algebra

- This is a first step towards categorification of harder algebraic theories. [1, 7]
- Quantum group theory can be formulated diagramatically [5]. What is e.g. sphericality, modularity, etc. of a Hopf pseudomonoid?
- Coherence results for Frobenius pseudomonoids — TQFT [3], surface foams [2].
1. **Background**
   - Result and motivation
   - **Full generality from semistrictness**
   - Introduction to semistrict bicategories: Gray monoids
   - Quasistrict braiding and symmetry

2. **Coherence for pseudomonoids**
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitors
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
What’s new: our results are proved in semistrict braided/symmetric monoidal bicategories

- Quasistrict [9] braided/symmetric monoidal bicategories are simpler than fully weak, but still:

**Lemma (Semistrictness of quasistrict braided and symmetric monoidal bicategories)**

The free weak braided/symmetric monoidal bicategory \( F_W(X) \) on a theory \( X \) is (braided/symmetric) monoidally biequivalent to \( F_Q(X) \), the free quasistrict braided/monoidal bicategory on that theory.

- The biequivalence identifies certain 1-morphisms and sends corresponding coherence 2-morphisms to the identity.
Semistrictness means our results apply in fully weak monoidal bicategories

- Algebraic theory generated by presentation $X$.
- An $X$-algebra in $C$ is a strict homomorphism $F_W(\Sigma_X) \rightarrow C$.
- Suppose we want to know if two 2-morphisms in $F_W(\Sigma_X)$ are equal.
  - Map along $F_W(\Sigma_X) \sim F_Q(\Sigma_X)$.
  - Biequivalences are faithful on 2-cells, so they are equal iff their images in $F_Q(\Sigma_X)$ are equal.
Outline

1 Background
   - Result and motivation
   - Full generality from semistrictness
   - Introduction to semistrict bicategories: Gray monoids
   - Quasistrict braiding and symmetry

2 Coherence for pseudomonoids
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitors
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
Gray monoids: ordered string diagrams

- To get Gray monoids we weaken planar isotopy of string diagrams in strict 2-categories.
- *Ordered string diagrams* are string diagrams where none of the generating 1-cells occur at the same vertical height. The equivalence relation is *ordered planar isotopy*, where string diagrams must remain ordered throughout the isotopy.
Generating 0-cells are string labels.
Generating 1-cells are boxes with specified input and output strings.
1-cells in $\mathcal{G}(C_0, C_1, C_2)$ are equivalence classes of ordered string diagrams under ordered planar isotopy.
The interchanger isomorphism

- Instead of equality of 1-cells, we now have an interchanger 2-isomorphism to control the isotopy:

\[ f \xrightarrow{\iota_{f,g}} g \xrightarrow{\iota_{f,g}} f \]

- The interchanger swaps the heights of two vertically adjacent unconnected 1-cells. Example:
2-cells in a Gray monoid

- Generating 2-cells have ordered string diagrams as source and target.
- 2-cells $D_1 \to D_n$ are sequences $D_1 \xrightarrow{\gamma_{1,2}} D_2 \xrightarrow{\gamma_{2,3}} \cdots \xrightarrow{\gamma_{n-1,n}} D_n$, where $D_i$ are 1-cells and $\gamma_{i,i+1}$ are either:
  - A generating 2-cell such that $D_i$ and $D_{i+1}$ differ only by the application of $\gamma_{i,i+1}$ on a rectangular subregion.
  - An *interchanger* 2-cell (next slide).
- If we draw them out, 2-cells are *movies* of ordered string diagrams.
- E.g. the associator:

\[ m : \]

\[ \begin{array}{c}
\xrightarrow{\alpha} \\
\xrightarrow{\alpha} \\
\xrightarrow{\alpha}
\end{array} \]
Equalities of 2-cells

- We can also specify certain equalities of 2-cells in our computad, provided they have the same source and target:

\[
[D_s \Rightarrow \gamma_{1,2} D_2 \Rightarrow \gamma_{2,3} \ldots \Rightarrow \gamma_{m-1,m} D_t] = [D_s \Rightarrow \delta_{1,2} \tilde{D}_2 \Rightarrow \delta_{2,3} \ldots \Rightarrow \delta_{n-1,n} D_t]
\]

- For example, the ‘pentagon’:

  \[
  \begin{bmatrix}
  \alpha & \Rightarrow & \alpha & \Rightarrow & \alpha \\
  \Rightarrow & \alpha & \Rightarrow & \alpha & \Rightarrow \\
  \Rightarrow & \Rightarrow & \Rightarrow & \Rightarrow
  \end{bmatrix}
  =
  \begin{bmatrix}
  \alpha & \Rightarrow & \iota & \Rightarrow & \alpha \\
  \Rightarrow & \Rightarrow & \Rightarrow & \Rightarrow
  \end{bmatrix}
  \]

- These generating equalities can be applied to contiguous subsequences on rectangular subregions.
Example: a rewrite of 2-cells in $\mathcal{G}(\mathcal{P})$
The projection of a 2-morphism

- Viewing a 2-morphism frame-by-frame can be unilluminating.
- At the cost of some information, we can view the whole 2-morphism in one planar diagram.
Revisiting the last example

[Diagram of a series of transformations involving graphs and arrows, indicating a process or algorithm.]
Structural equalities of a Gray monoid: Type I

- If two 2-cells on disjoint rectangular subregions occur one after the other in a movie, we can exchange their order.

- This corresponds to an interchanger in the projection.
Structural equalities of a Gray monoid: Type II

- We can insert 2-cell and its inverse at any frame containing a rectangular subregion with its source. We call this an *inverse insert*.
- Going the other way is called *cancellation*.

\[
\begin{align*}
\begin{bmatrix}
\text{[ ]} & \text{[ ]} & \Rightarrow & \text{[ ]} & \Rightarrow & \text{[ ]}
\end{bmatrix}
\quad \Rightarrow \\
\begin{bmatrix}
\text{[ ]} & \text{[ ]} & \Rightarrow & \text{[ ]} & \Rightarrow & \text{[ ]}
\end{bmatrix}
\end{align*}
\]
Structural equalities of a Gray monoid: Type III

- Interchanging with all the nodes in a 2-cell, then performing the 2-cell, is equal to performing the 2-cell and then interchanging.

- In the projection, this is a pullthrough:
Outline

1. **Background**
   - Result and motivation
   - Full generality from semistrictness
   - Introduction to semistrict bicategories: Gray monoids
   - Quasistrict braiding and symmetry

2. **Coherence for pseudomonoids**
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitors
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
A quasistrict braided Gray monoid has an additional four ‘braiding’ 1-cells for every pair $A, B$ of generating 0-cells:

Braidings on products of generating 0-cells are compositions, e.g. $\sigma_{A \otimes B, C} = (\sigma_{A \otimes C} \otimes \mathbb{1}_B) \circ (\mathbb{1}_A \otimes \sigma_{B \otimes C})$: 
Quasistrict braided structure: II

- Braidings obey the following strict equalities:
The symmetric structure [9] is exactly the same as the braided structure, except with the additional equality:
The braided and symmetric monoidal bicategories in Globular are roughly *Crans* type [3] - weaker than quasistrict.

In the full paper [10], we use Crans axioms.

Requires use of simple normalisation routine (‘top string normal form’) and additional case checking.
Outline

1. Background
   - Result and motivation
   - Full generality from semistrictness
   - Introduction to semistrict bicategories: Gray monoids
   - Quasistrict braiding and symmetry

2. Coherence for pseudomonoids
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitors
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
Example: The pseudomonoid computad

- 0-cells: \{ C \}.
- 1-cells: \( m : C \times C \to C \) and \( u : I \to C \).
  \[
  m = \quad u =
  \]
- 2-cells: \( \alpha \) (associator), \( \lambda \) (left unitor) and \( \rho \) (right unitor), all isomorphisms.
The equalities of the pseudomonoid computad

Pentagon

Triangle
Braided pseudomonoids

- Extra *commutator* 2-isomorphism:

- Hexagon equality 1:

  \[
  \begin{bmatrix}
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \end{bmatrix}
  =
  \begin{bmatrix}
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha^{-1}} \\
  \xrightarrow{c}
  \end{array}
  \end{bmatrix}
  \]

- Hexagon equality 2:

  \[
  \begin{bmatrix}
  \begin{array}{c}
  \xrightarrow{\alpha} \\
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \end{bmatrix}
  =
  \begin{bmatrix}
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \\
  \begin{array}{c}
  \xrightarrow{c} \\
  \xrightarrow{\alpha}
  \end{array}
  \end{bmatrix}
  \]

Symmetric pseudomonoids

Symmetry equality:

\[
\begin{bmatrix}
\begin{array}{c}
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\end{array}
\end{bmatrix}
= 
\begin{bmatrix}
\begin{array}{c}
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\end{array}
\end{bmatrix}
= 
\begin{bmatrix}
\begin{array}{c}
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\end{array}
\end{bmatrix}
\cong
\begin{bmatrix}
\begin{array}{c}
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\end{array}
\end{bmatrix}
\cong
\begin{bmatrix}
\begin{array}{c}
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\cdot \\
\end{array}
\end{bmatrix}
\]
1 Background
- Result and motivation
- Full generality from semistrictness
- Introduction to semistrict bicategories: Gray monoids
- Quasistrict braiding and symmetry

2 Coherence for pseudomonoids
- Presentations of pseudomonoids
- An overview of the proof
  - Proof step 1: Removing unitors
  - Proof step 2: Fixing trees
  - Proof step 3: Showing braid relations
In this presentation we consider braided pseudomonoids in symmetric monoidal bicategories.

In the paper we show a symmetric monoidal biequivalence between the free symmetric Gray monoid on the braided pseudomonoid computad and a certain strict symmetric monoidal bicategory.

Here we ignore the compositional structure and show two consequences:

- A ‘normal form’ for 2-morphisms.
- A solution to the word problem for 2-morphisms.
In the PROP for commutative monoids, the associator and unitors are strict equalities.

There is a normal form for 1-cells. Take any 1-cell:

Pull all the units upwards and eliminate them using the unitor equalities if attached.

Pull the multiplications up above the braidings...

Then use commutators and associators to left bracket the trees, with ordered inputs.
In the PROP for commutative monoids, the associator and unitors are strict equalities.

There is a normal form for 1-cells. Take any 1-cell:

- Pull all the units upwards and eliminate them using the unitor equalities if attached.
- Pull the multiplications up above the braidings...
- Then use commutators and associators to left bracket the trees, with ordered inputs.
In the PROP for commutative monoids, the associator and unitors are strict equalities.

There is a normal form for 1-cells. Take any 1-cell:

- Pull all the units upwards and eliminate them using the unitor equalities if attached.
- Pull the multiplications up above the braidings...
- Then use commutators and associators to left bracket the trees, with ordered inputs.
In the PROP for commutative monoids, the associator and unitors are strict equalities.

There is a normal form for 1-cells. Take any 1-cell:

- Pull all the units upwards and eliminate them using the unitor equalities if attached.
- Pull the multiplications up above the braidings...
- Then use commutators and associators to left bracket the trees, with ordered inputs.
In the PROP for commutative monoids, the associator and unitors are strict equalities.

There is a normal form for 1-cells. Take any 1-cell:

- Pull all the units upwards and eliminate them using the unitor equalities if attached.
- Pull the multiplications up above the braidings...
- Then use commutators and associators to left bracket the trees, with ordered inputs.
In the PROP for commutative monoids, the associator and unitors are strict equalities.

There is a normal form for 1-cells. Take any 1-cell:

- Pull all the units upwards and eliminate them using the unitor equalities if attached.
- Pull the multiplications up above the braidings...
- Then use commutators and associators to left bracket the trees, with ordered inputs.
The PROP for commutative monoids

- In the PROP for commutative monoids, the associator and unitors are strict equalities.
- There is a normal form for 1-cells. Take any 1-cell:

Pull all the units upwards and eliminate them using the unitor equalities if attached.
Pull the multiplications up above the braidings...
Then use commutators and associators to left bracket the trees, with ordered inputs.
The word problem for the commutative monoid PROP, solved by a normal form

- A 1-cell in this normal form corresponds exactly to a function \( \{1, \ldots, m\} \to \{1, \ldots, n\} \), which it is easy to read off [8].

- All the equalities in the commutative monoid PROP preserve this function.

- So this normal form solves the word problem — every 1-cell \( f \) is equal to some \( N_f \), and obviously \( f = f' \) iff \( N_f = N_{f'} \).
Going up to 2 dimensions

- This equality in the 1D case becomes an isomorphism in the 2D case.
- This means that all 1-cells are in the isomorphism class of exactly one 1-cell in this standard form.
- Moreover, two one-cells $f, f'$ are isomorphic iff $N_f = N_{f'}$.
- We want to know when two 2-cells $\alpha, \beta : f \to g$ are equal.
Everything is an isomorphism.

So the result for 1-morphisms means that to solve the equality problem for 2-morphisms, we need only solve it for loops on 1-morphisms in normal form:

\[ \alpha = \beta \text{ iff } \nu^{-1} \circ \beta^{-1} \circ \alpha \circ \nu = id_{N_{f}}. \]
To solve the equality problem for 2-loops $\alpha : N_f \rightarrow N_f$ on normal form 1-cells $N_f$, we find a normal form for such loops, such that every $\alpha : N_f \rightarrow N_f$ is equal to some unique $N_{\alpha} : N_f \rightarrow N_f$.

We now specify this normal form.
Normal form for 2-loops on normal form 1-cells

Permutation of inputs

Step 2n.

Step 2n − 1.

σn

σ1
Normal form and equality

- One normal form loop for each choice of product pure braid group element $\sigma_1 \times \cdots \times \sigma_n$ to be absorbed.
- None of the equalities of the braided pseudomonoid (pentagon, triangle, hexagons) change the braid group element absorbed — so none of these normal form loops is equal.
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
We are presented with some loop on a normal form 1-cell. Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural
  equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
The task of rewriting a given loop

- We are presented with some loop on a normal form 1-cell.
- Our task: rewrite it into normal form using structural equalities, pentagon, triangle and hexagons. For example:
Outline

1 Background
   - Result and motivation
   - Full generality from semistrictness
   - Introduction to semistrict bicategories: Gray monoids
   - Quasistrict braiding and symmetry

2 Coherence for pseudomonoids
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitors
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
Notice that there are no attached units on 1-cells in normal form.

An inverse unitor creates an attached unit.

The attached unit can’t be made unattached using any of the 2-cells.

Since the movie ends on a 1-cell in normal form, the attached unit must be removed by a unitor.

So every inverse unitor is paired with a unitor which removes the created unit.

Idea: move the inverse unitor back to the end of the movie, so that it meets the unitor and so may be cancelled.
Consider the last inverse unitor in the loop.

Suppose that no 2-cell affects the created multiplication node throughout the loop, apart from the unitor which removes it.

We will now show how to cancel the unitor with its inverse.
Because the braiding in the quasistrict category is trivial, the unit and multiplication will form a loop in the projection unentangled with the other morphisms.

Push the inverse unitor up using Type I rewrites.

If this becomes impossible, it will be due to a chain of downwards interchangers (it can’t be another inverse unitor, since this is the last one).

Go to the end of the chain of downwards interchangers and push the last one back using Type I rewrites.

If this is impossible it will be due to an upwards interchanger or a 2-cell acting on the 1-cell with which the unit was interchanged; this may be pulled through.

Repeating this, we remove the loop.
Easy case: Example

In the projection:
Reducing to the easy case: I

- It may be that there is some 2-cell other than the unitor which acts on the last created multiplication node.
Reducing to the easy case: I

- In this case, use Type II rewrites to insert interchangers and their inverses immediately before the 2-cell so that the unit node goes straight up to the multiplication, returns and then the 2-cell occurs.
- Use Type I rewrites to bring the 2-cell before the pulldowns.
- Insert a unit destruction operator and its inverse immediately before the 2-cell.
- Eliminate the first unit creation operator and the inserted destruction operator (Case I).
- Now use Type I rewrites so that the 2-cell happens immediately after the unit creation.
Reducing to the easy case: I (Example)

- In the projection:

- So now we reduced to the case where the 2-cell happens straight after the inverse unitor.
Reducing to the easy case: II

- Now we must consider cases individually.
- We consider each 2-cell that can occur on the multiplication node immediately after the unitor.
- We show that there is always some rewrite that ‘pulls the unitor through’ the 2-cell.
- We show some examples.
Example I: Multiplication node lower partner in associator

\[
\begin{align*}
\left[ \begin{array}{c}
\text{upper node} \\
\text{lower node}
\end{array} \right] & \Rightarrow \left[ \begin{array}{c}
\text{upper node} \\
\text{lower node}
\end{array} \right] & = & \left[ \begin{array}{c}
\text{upper node} \\
\text{lower node}
\end{array} \right] & \Rightarrow \left[ \begin{array}{c}
\text{upper node} \\
\text{lower node}
\end{array} \right] & \Rightarrow \left[ \begin{array}{c}
\text{upper node} \\
\text{lower node}
\end{array} \right]
\end{align*}
\]
Example 2: Commutator acts on multiplication node

\[
\begin{bmatrix}
  \quad\quad \quad \quad \quad \quad \\
\end{bmatrix} \Rightarrow
\begin{bmatrix}
  \quad\quad \quad \quad \quad \quad \\
\end{bmatrix} \Rightarrow
\begin{bmatrix}
  \quad\quad \quad \quad \quad \quad \\
\end{bmatrix} \Rightarrow
\begin{bmatrix}
  \quad\quad \quad \quad \quad \quad \\
\end{bmatrix} =
\begin{bmatrix}
  \quad\quad \quad \quad \quad \quad \\
\end{bmatrix} \Rightarrow
\begin{bmatrix}
  \quad\quad \quad \quad \quad \quad \\
\end{bmatrix}
\]
By using these rewrites for all the 2-cells on the multiplication node, we reduce to the easy case. We then cancel the last inverse unitor in the loop. Iterating, we remove all the unitors/inverse unitors in the loop.
Outline

1 Background
   - Result and motivation
   - Full generality from semistrictness
   - Introduction to semistrict bicategories: Gray monoids
   - Quasistrict braiding and symmetry

2 Coherence for pseudomonoids
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitor relations
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
Fixing trees: the idea

- The next step is to get the loop into the following form:

- In the red region, only associators occur. In the grey region, only commutators. In the green region, nothing happens (because the braid relations are equalities).
- This uses the pentagon and hexagon equalities.
Outline

1. Background
   - Result and motivation
   - Full generality from semistrictness
   - Introduction to semistrict bicategories: Gray monoids
   - Quasistrict braiding and symmetry

2. Coherence for pseudomonoids
   - Presentations of pseudomonoids
   - An overview of the proof
   - Proof step 1: Removing unitors
   - Proof step 2: Fixing trees
   - Proof step 3: Showing braid relations
The movie is in the above form, where some braid is absorbed.

Recall that our normal form distinguished only between isotopy classes of pure braids absorbed.

Right now *any* word in the generators of the braid group might be absorbed.

So, we need to show that the movies can be rewritten using the relations in the braid group.
Two easy properties

- Associativity of composition is easy, because in the Gray monoid associativity holds strictly.
- For inverses, just perform a cancellation.
We need to show that $\sigma_i \sigma_j = \sigma_j \sigma_i$ for $|i - j| > 1$. 

[Diagram of braid relations and transformations]
Braid relation 1: Proof
Braid relation 1: Proof
We need to show that $\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}$.

\[
\begin{bmatrix}
\text{Diagram 1}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 2}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 3}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 4}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 5}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 6}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 7}
\end{bmatrix}
\]

\[
= \begin{bmatrix}
\text{Diagram 8}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 9}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 10}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 11}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 12}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 13}
\end{bmatrix} \Rightarrow \begin{bmatrix}
\text{Diagram 14}
\end{bmatrix}
\]
Braid relation 2: Proof
Braid relation 2: Proof

⇒

→

→

→

→
So, we rewrote each loop to a normal form loop. All the normal form loops are non-equal. So this solves the word problem. In the paper we use this to build a symmetric monoidal biequivalence with a symmetric monoidal 2-category constructed from the data of normal form loops.
We prove this using weaker Crans axioms.
We prove biequivalences.
We prove similar results for all possible combinations:
- Pseudomonoids in monoidal, braided monoidal and symmetric monoidal bicategories.
- Braided pseudomonoids in braided monoidal and symmetric monoidal bicategories.
- Symmetric pseudomonoids in symmetric monoidal bicategories.
Summary

- We can do higher algebra with string diagrams in the weak case.
- Quasistrict bicategories and *Globular* make things a lot easier.

Outlook

- Next stop: pseudobialgebras, pseudo-Hopf algebras
- Can this proof be directed so as to use rewriting techniques [6]?

Thanks for listening!
References I


Martin Neuchl.  
*Representation theory of Hopf categories.*  

Teimuraz Pirashvili.  
On the PROP corresponding to bialgebras.  

Christopher John Schommer-Pries.  
*The classification of two-dimensional extended topological field theories.*  
Dominic Verdon.  
Coherence for braided and symmetric pseudomonoids.  