Traced concategories

Paul Blain Levy, Sergey Goncharov and Lutz Schröder

December 18, 2018

Outline

Varying the notion of category

Notion	Morphism	Main example
Category	$f : a \to b$	

Varying the notion of category

Notion	Morphism	Main example
Category	$f: a \to b$	
Cartesian multicategory	$f: \overrightarrow{a} \to b$	$f: \prod \overrightarrow{a} \to b$ in a cartesian category

Varying the notion of category

Notion	Morphism	Main example
Category	$f\colon a \to b$	
Cartesian multicategory	$f\colon \overrightarrow{a} \to b$	$f\colon \prod \overrightarrow{a} \to b$ in a cartesian category
Multicategory	$f\colon \overrightarrow{a} \to b$	$f\colon \bigotimes \overrightarrow{a} \to b$ in a monoidal category

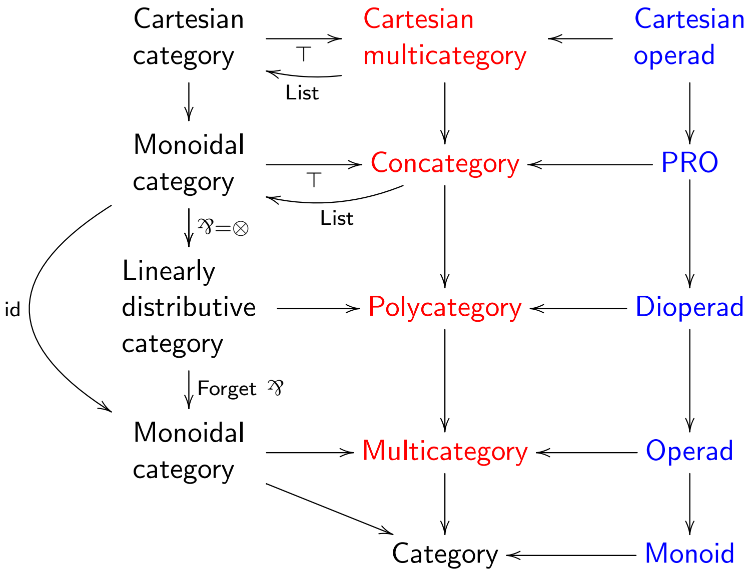
Varying the notion of category

Notion	Morphism	Main example
Category	$f\colon a \to b$	
Cartesian multicategory	$f\colon \overrightarrow{a} \to b$	$f\colon \prod \overrightarrow{a} \to b$ in a cartesian category
Multicategory	$f\colon \overrightarrow{a} \to b$	$f\colon \bigotimes \overrightarrow{a} \to b$ in a monoidal category
Concategory	$f\colon \overrightarrow{a} \to \overrightarrow{b}$	$f\colon \bigotimes \overrightarrow{a} \to \bigotimes \overrightarrow{b}$ in a monoidal category

Varying the notion of category

Notion	Morphism	Main example
Category	$f : a \to b$	
Cartesian multicategory	$f : \overrightarrow{a} \to b$	$f : \prod \overrightarrow{a} \to b$ in a cartesian category
Multicategory	$f : \overrightarrow{a} \to b$	$f : \bigotimes \overrightarrow{a} \to b$ in a monoidal category
Concategory	$f : \overrightarrow{a} \to \overrightarrow{b}$	$f : \bigotimes \overrightarrow{a} \to \bigotimes \overrightarrow{b}$ in a monoidal category
Polycategory	$f : \overrightarrow{a} \to \overrightarrow{b}$	$f : \bigotimes \overrightarrow{a} \to \invamp \overrightarrow{b}$ in a linearly distributive category

Constructions



Definition of concategory

A *concategory* $\mathcal{C}$ consists of the following data.

- A class ob $\mathcal{C}$ of objects.
- A homset $\mathcal{C}(\overrightarrow{a}; \overrightarrow{b})$ for each pair of object lists $\overrightarrow{a}, \overrightarrow{b}$.

Definition of concategory

A *concategory* $\mathcal{C}$ consists of the following data.

- A class ob $\mathcal{C}$ of objects.
- A homset $\mathcal{C}(\overrightarrow{a}; \overrightarrow{b})$ for each pair of object lists $\overrightarrow{a}, \overrightarrow{b}$.
- The *sequential composite* of $f \colon \overrightarrow{a} \to \overrightarrow{b}$ and $g \colon \overrightarrow{b} \to \overrightarrow{c}$ is $f; g \colon \overrightarrow{a} \to \overrightarrow{c}$.
- The *parallel composite* of $f \colon \overrightarrow{a} \to \overrightarrow{b}$ and $g \colon \overrightarrow{c} \to \overrightarrow{d}$ is $f \boxtimes g \colon \overrightarrow{a} +\!\!+ \overrightarrow{c} \to \overrightarrow{b} +\!\!+ \overrightarrow{d}$.

Definition of concategory

A *concategory* $\mathcal{C}$ consists of the following data.

- A class ob $\mathcal{C}$ of objects.
- A homset $\mathcal{C}(\overrightarrow{a}; \overrightarrow{b})$ for each pair of object lists $\overrightarrow{a}, \overrightarrow{b}$.
- The *sequential composite* of $f \colon \overrightarrow{a} \to \overrightarrow{b}$ and $g \colon \overrightarrow{b} \to \overrightarrow{c}$ is $f ; g \colon \overrightarrow{a} \to \overrightarrow{c}$.
- The *parallel composite* of $f \colon \overrightarrow{a} \to \overrightarrow{b}$ and $g \colon \overrightarrow{c} \to \overrightarrow{d}$ is $f \boxtimes g \colon \overrightarrow{a} + \!\!+ \overrightarrow{c} \to \overrightarrow{b} + \!\!+ \overrightarrow{d}$.
- The sequential identity $\mathrm{id}_{\overrightarrow{a}} \colon \overrightarrow{a} \to \overrightarrow{a}$.
- The parallel identity $\mathrm{id}_{\boxtimes} \colon \varepsilon \to \varepsilon$. (Redundant.)

The ten commandments

- Sequential composition is associative and unital.
- Parallel composition is associative and unital.
- Interchange between sequential and parallel composition:

$$(f; g) \boxtimes (h; k) \quad = \quad (f \boxtimes h); (g \boxtimes k)$$

- Interchange between sequential identity and parallel composition:

$$\mathsf{id}_{\vec{a}} \boxtimes \mathsf{id}_{\vec{b}} \quad = \quad \mathsf{id}_{\vec{a} + \!\!\! + \vec{b}}$$

- Interchange between sequential composition and parallel identity:

$$\mathsf{id}_{\boxtimes} \quad = \quad \mathsf{id}_{\boxtimes}; \mathsf{id}_{\boxtimes}$$

- Interchange between sequential and parallel identity:

$$\mathsf{id}_{\varepsilon} \quad = \quad \mathsf{id}_{\boxtimes}$$

Why the name?

- "Category" alludes to sequential composition

$$f;g\colon \overrightarrow{a} \to \overrightarrow{c}$$

- "Concat" alludes to parallel composition

$$f \boxtimes g\colon \overrightarrow{a} + \overrightarrow{c} \to \overrightarrow{b} + \overrightarrow{d}$$

Why the name?

- "Category" alludes to sequential composition

$$f ; g \colon \overrightarrow{a} \to \overrightarrow{c}$$

- "Concat" alludes to parallel composition

$$f \boxtimes g \colon \overrightarrow{a} + \overrightarrow{c} \to \overrightarrow{b} + \overrightarrow{d}$$

- The overlap alludes to the interchange law.

Caviglia's 2-category of concategories

Map of concategories

A map $F\colon \mathcal{C} \to \mathcal{D}$ sends objects to objects and morphisms to morphisms, preserving all structure.

Natural transformation

A natural transformation sends each object a to $\alpha_a\colon [Fa] \to [Ga]$.

For $f\colon \overrightarrow{a} \to \overrightarrow{b}$ we require $f; \overrightarrow{\alpha_b} = \overrightarrow{\alpha_a}; f$.

Examples of concategories

Each of the following gives a concategory:

1. A monoidal category.
   Morphisms go from $\bigotimes \overrightarrow{a} \to \bigotimes \overrightarrow{b}$.

Examples of concategories

Each of the following gives a concategory:

1. A monoidal category.
   Morphisms go from $\bigotimes \overrightarrow{a} \to \bigotimes \overrightarrow{b}$.
2. A cartesian multicategory.

Examples of concategories

Each of the following gives a concategory:

1. A monoidal category.
   Morphisms go from $\bigotimes \overrightarrow{a} \to \bigotimes \overrightarrow{b}$.
2. A cartesian multicategory.
3. A PRO.
   It corresponds to a single-object concategory

Examples of concategories

Each of the following gives a concategory:

1. A monoidal category.
   Morphisms go from $\bigotimes \overrightarrow{a} \to \bigotimes \overrightarrow{b}$.
2. A cartesian multicategory.
3. A PRO.
   It corresponds to a single-object concategory
4. A many-sorted list-to-list signature.
   Acyclic string diagrams modulo isomorphism.

Examples of concategories

Each of the following gives a concategory:

1. A monoidal category.
   Morphisms go from $\bigotimes \overrightarrow{a} \to \bigotimes \overrightarrow{b}$.
2. A cartesian multicategory.
3. A PRO.
   It corresponds to a single-object concategory
4. A many-sorted list-to-list signature.
   Acyclic string diagrams modulo isomorphism.
5. A dataflow model e.g. Kahn's or Jonsson's.

PROs

- A PRO consists of a family of sets $(A_{m,n})_{m,n\in\mathbb{N}}$

  with $f \in A_{m,n}$ written $f\colon m \to n$

  and sequential and parallel composition and identity

  satisfying the ten commandments.
- A PRO A correspond to a single-object concategory $\tilde{A}$.

Colours

Object = "colour".

Concategory = "coloured PRO"

Multicategory = "coloured operad"

Polycategory = "coloured dioperad"

Colours

Object = "colour".

Concategory = "coloured PRO"

Multicategory = "coloured operad"

Polycategory = "coloured dioperad"

Category = "coloured monoid" (Tom Leinster satire)

Colours

Object = “colour”.

Concategory = “coloured PRO”

Multicategory = “coloured operad”

Polycategory = “coloured dioperad”

Category = “coloured monoid” (Tom Leinster satire)

Monoidal category = “monoidal coloured monoid”

Colours

Object = "colour".

Concategory = "coloured PRO"

Multicategory = "coloured operad"

Polycategory = "coloured dioperad"

Category = "coloured monoid" (Tom Leinster satire)

Monoidal category = "monoidal coloured monoid"

In colourful literature, usually:

- Colours form a set, sometimes a finite set, sometimes fixed in advance.
- The construction monoidal category $\rightarrow$ concategory is not prominent.

Concategory vs monoidal category

The 2-embedding of $\mathbf{MONCAT}$ in $\mathbf{CONCAT}$ is reflective.

$$\mathbf{MONCAT} \underset{\mathsf{List}}{\overset{\top}{\rightleftarrows}} \mathbf{CONCAT}$$

$\mathsf{List}\,\mathcal{C}$ is a strict monoidal category.

Its objects are lists of $\mathcal{C}$-objects.

The induced comonad on $\mathbf{MONCAT}$ is strictification.

So we have resolved strictification into two parts.

Is a concategory a strict monoidal category?

Here are two concategories:

- the PRO of complex matrices, regarded as a concategory
- the monoidal category of finite dimensional Hilbert spaces with $\oplus$, regarded as a concategory.

They are not equivalent concategories,
but List sends them to equivalent strict monoidal categories.

Symmetric concategory

Given a morphism $f\colon \overrightarrow{a} \to \overrightarrow{b}$

a pre-symmetry allows you to swap two adjacent wires into f

or two adjacent wires out of f

with suitable laws.

This gives actions of the symmetric group.

Given a morphism $f\colon \overrightarrow{a} \to \overrightarrow{b}$

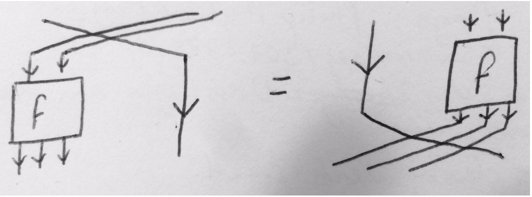
a pre-symmetry allows you to swap two adjacent wires into f

or two adjacent wires out of f

with suitable laws.

This gives actions of the symmetric group.

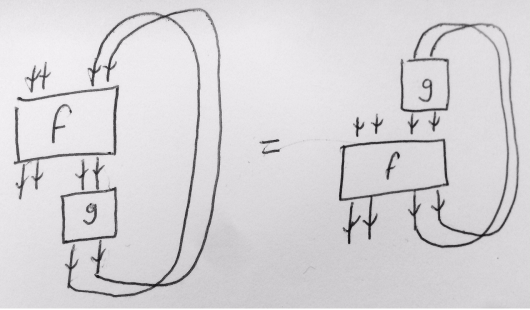
The pre-symmetry is a symmetry when we have the naturality law:



A PRO with symmetry is called a PROP.

Traced concategory

- A pre-trace for a symmetric concategory takes a morphism $f\colon \overrightarrow{a}, c \to \overrightarrow{b}, c$ to a morphism $f\colon \overrightarrow{a} \to \overrightarrow{b}$.
- Must be natural in $\overrightarrow{a}$ and $\overrightarrow{b}$ and satisfy vanishing I, vanishing II, superposing and yanking.
- Then a morphism $f\colon \overrightarrow{a}, \overrightarrow{c} \to \overrightarrow{b}, \overrightarrow{c}$ gives a morphism $f\colon \overrightarrow{a} \to \overrightarrow{b}$.
- The pre-trace is a trace when this is dinatural in $\overrightarrow{c}$.



String diagrams

- A many-sorted list-to-list signature $\mathcal{S}$ is
  - a set of sorts
  - a set of symbols equipped with a pair of lists of sorts.
- A string diagram on $\mathcal{S}$ consists of
  - a set of boxes, each assigned a symbol
  - a bijection from the output ports to the input ports.
- String diagrams modulo isomorphism is the free traced concategory on $\mathcal{S}$. (To be checked)

String diagrams

- A many-sorted list-to-list signature $\mathcal{S}$ is
  - a set of sorts
  - a set of symbols equipped with a pair of lists of sorts.
- A string diagram on $\mathcal{S}$ consists of
  - a set of boxes, each assigned a symbol
  - a bijection from the output ports to the input ports.
- String diagrams modulo isomorphism is the free traced concategory on $\mathcal{S}$. (To be checked) (or confirmed by audience)

String diagrams

- A many-sorted list-to-list signature $\mathcal{S}$ is
  - a set of sorts
  - a set of symbols equipped with a pair of lists of sorts.
- A string diagram on $\mathcal{S}$ consists of
  - a set of boxes, each assigned a symbol
  - a bijection from the output ports to the input ports.
- String diagrams modulo isomorphism is the free traced concategory on $\mathcal{S}$. (To be checked) (or confirmed by audience)
- Acyclic string diagrams modulo isomorphism is the free symmetric concategory on $\mathcal{S}$.

Sermon

Often said

- “String diagrams are a great notation for monoidal categories.”

Sermon

Often said

- “String diagrams are a great notation for monoidal categories.”

Sometimes said

- “String diagrams are a great notation for strict monoidal categories.”

Sermon

Often said

- "String diagrams are a great notation for monoidal categories."

Sometimes said

- "String diagrams are a great notation for strict monoidal categories."

The truth is in between

- "String diagrams are a great notation for concategories."

More sermons

Monoids

In a monoidal category.

More generally in a multicategory.

More sermons

Monoids

In a monoidal category.

More generally in a multicategory.

Bimonoids

In a symmetric monoidal category.

More generally in a symmetric concategory.

More sermons

Monoids

In a monoidal category.

More generally in a multicategory.

Bimonoids

In a symmetric monoidal category.

More generally in a symmetric concategory.

Dual objects

In a symmetric monoidal category.

More generally in a symmetric polycategory.

More sermons

Monoids

In a monoidal category.

More generally in a multicategory.

Bimonoids

In a symmetric monoidal category.

More generally in a symmetric concategory.

Dual objects

In a symmetric monoidal category.

More generally in a symmetric polycategory.

Models of a cartesian operad (equivalently, Lawvere theory)

In a cartesian category.

More generally in a cartesian multicategory.

More sermons

Monoids

In a monoidal category.

More generally in a multicategory.

Bimonoids

In a symmetric monoidal category.

More generally in a symmetric concategory.

Dual objects

In a symmetric monoidal category.

More generally in a symmetric polycategory.

Models of a cartesian operad (equivalently, Lawvere theory)

In a cartesian category.

More generally in a cartesian multicategory.

Models of a PROP

In a symmetric monoidal category.

More generally in a symmetric concategory.

Dataflow models

- A dataflow program inputs data from several channels and outputs data to several channels.
- Each channel has a set of permitted values.

Dataflow models

- A dataflow program inputs data from several channels and outputs data to several channels.
- Each channel has a set of permitted values.
- Kahn gave a model of deterministic dataflow.
- Jonsson gave a model of nondeterministic dataflow.

Dataflow models

- A dataflow program inputs data from several channels and outputs data to several channels.
- Each channel has a set of permitted values.
- Kahn gave a model of deterministic dataflow.
- Jonsson gave a model of nondeterministic dataflow.
- These form traced concategories. (To be checked.)

Kahn's dataflow example

Objects are sets.

Stream(A) is the domain of finite and infinite streams of values in A.

A morphism from $(A_i)_{i<m}$ to $(B_j)_{j<n}$ is a continuous function

$$\prod_{i<m} \mathsf{Stream}(A_i) \to \prod_{j<n} \mathsf{Stream}(B_j)$$

Kahn's dataflow example

Objects are sets.

Stream(A) is the domain of finite and infinite streams of values in A.

A morphism from $(A_i)_{i<m}$ to $(B_j)_{j<n}$ is a continuous function

$$\prod_{i<m} \mathsf{Stream}(A_i) \to \prod_{j<n} \mathsf{Stream}(B_j)$$

Trace is least (pre)fixpoint.

Further work

- Lots of expected things need to be checked.
- Guarded traces? (Goncharov and Schröder, FoSSaCS 2018)