A Syntactical Definition of Opetopes

Pierre-Louis Curien[1] Cédric Ho Thanh[2] Samuel Mimram[3]

πr^2 "mise au vert", April 11, 2018

[1] IRIF, Paris Diderot University

[2] IRIF, Paris Diderot University, INSPIRE 2017 Fellow, This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 665850

[3] LIX, École Polytechnique

Opetopes

Definition: in the shell of a very small nut

Opetopes are shapes (akin to globules, cubes, simplices, etc.), designed to represent the notion of composition in every dimension. As such, they were introduced to describe laws and coherence in higher categories.

Definition: in the shell of a very small nut

Opetopes are shapes (akin to globules, cubes, simplices, etc.), designed to represent the notion of composition in every dimension. As such, they were introduced to describe laws and coherence in higher categories.

Their distinctive feature is that they are *many-to-one*, and thus exhibit a tree-like structure.

Definition: base cases

- There is a unique 0-dimensional opetope, which we'll call the *point* and denote by ♦.

Definition: base cases

- There is a unique 0-dimensional opetope, which we'll call the *point* and denote by $\blacklozenge$.
- There is a unique 1-opetope, the arrow, denoted by $\blacksquare$.

Definition: base cases

- There is a unique 0-dimensional opetope, which we'll call the *point* and denote by $\blacklozenge$.
- There is a unique 1-opetope, the arrow, denoted by $\blacksquare$. The arrow has a *source* $\mathsf{s}\,\blacksquare = \blacklozenge$ and a *target* $\mathsf{t}\,\blacksquare = \blacklozenge$.

$$\cdot \longrightarrow \cdot$$

Definition: dimension 2

The opetopes ♦ and ▪ can be arranged in a tree (corolla) as follows:

$$\cdot \longrightarrow \cdot$$

Definition: dimension 2

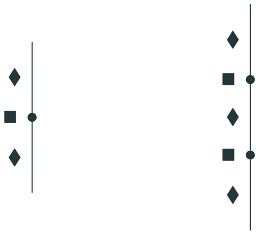
A 2-*opetope* is then a coherent tree made with this corolla.

Definition: dimension 2

A 2-*opetope* is then a coherent tree made with this corolla.

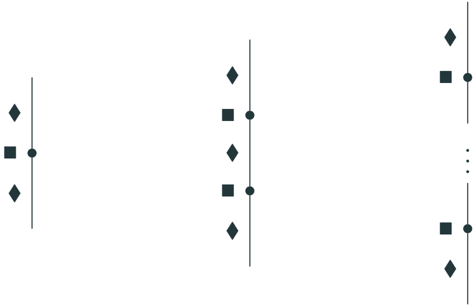
Definition: dimension 2

A 2-*opetope* is then a coherent tree made with this corolla.



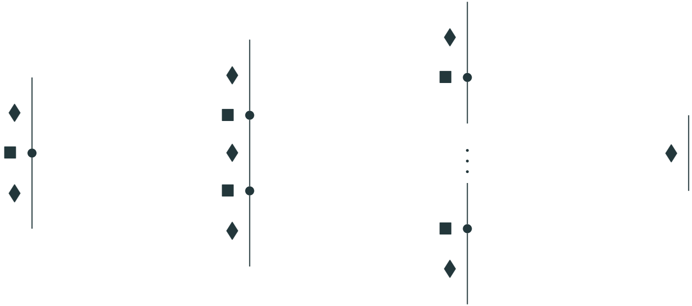
Definition: dimension 2

A 2-*opetope* is then a coherent tree made with this corolla.



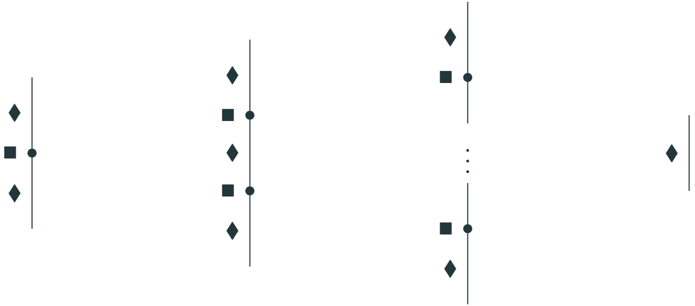
Definition: dimension 2

A 2-*opetope* is then a coherent tree made with this corolla.



Definition: dimension 2

A 2-*opetope* is then a coherent tree made with this corolla.



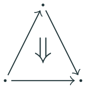
Since the input edge and output edge of the above corolla have the same decoration (♦), the coherence condition is trivially satisfied.

Definition: dimension 2

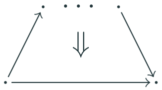
Intuitively, such trees tell us how to compose the 1-opetope ■ with itself.

Definition: dimension 2

Intuitively, such trees tell us how to compose the 1-opetope ■ with itself.

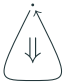


Intuitively, such trees tell us how to compose the 1-opetope ■ with itself.



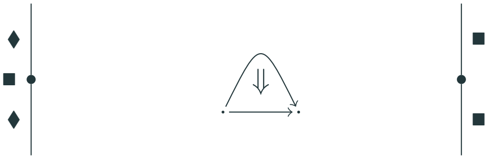
Definition: dimension 2

Intuitively, such trees tell us how to compose the 1-opetope ■ with itself.



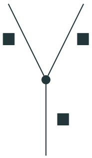
Definition: dimension 2

In turn, a 2-opetope can be seen as a corolla, whose input vertices correspond to the occurrences of 1-opetopes (or equivalently, to its nodes). The 1-opetopes decorating the input edges are called the *sources*, while that decorating the root edge is the *target*.



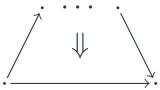
Definition: dimension 2

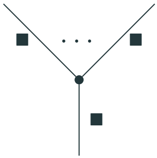
In turn, a 2-opetope can be seen as a corolla, whose input vertices correspond to the occurrences of 1-opetopes (or equivalently, to its nodes). The 1-opetopes decorating the input edges are called the *sources*, while that decorating the root edge is the *target*.



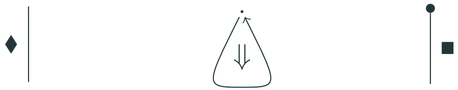
Definition: dimension 2

In turn, a 2-opetope can be seen as a corolla, whose input vertices correspond to the occurrences of 1-opetopes (or equivalently, to its nodes). The 1-opetopes decorating the input edges are called the *sources*, while that decorating the root edge is the *target*.



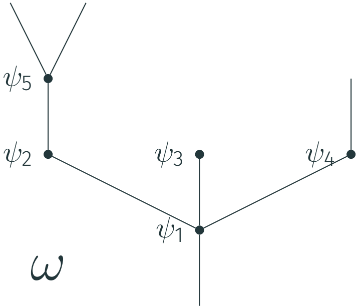


In turn, a 2-opetope can be seen as a corolla, whose input vertices correspond to the occurrences of 1-opetopes (or equivalently, to its nodes). The 1-opetopes decorating the input edges are called the *sources*, while that decorating the root edge is the *target*.

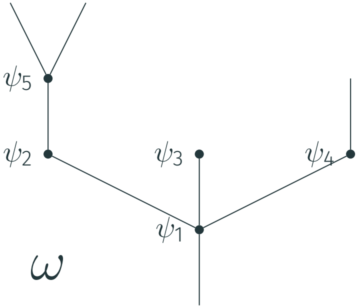


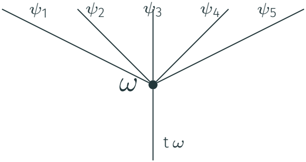
Definition: dimension ≥ 2

For $n \geq 2$, an n-opetope is a tree made from the corollas associated to $(n-1)$-opetopes.



For $n \geq 2$, an n-opetope is a tree made from the corollas associated to $(n-1)$-opetopes.





In turn, it can be seen as a corolla whose input vertices correspond to the occurrences of $(n-1)$-opetopes.

Definition: dimension ≥ 2

Equivalently, for $n \geq 2$:

Definition

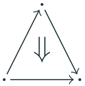
An n-opetope is a (finite) tree whose

- nodes are decorated by $(n-1)$-opetopes,
- edges are decorated by $(n-2)$-opetopes,
- all in a coherent way.

Graphical representations

Pasting schemes

As we saw before, a seemingly natural way of representing opetopes is by pasting schemes.

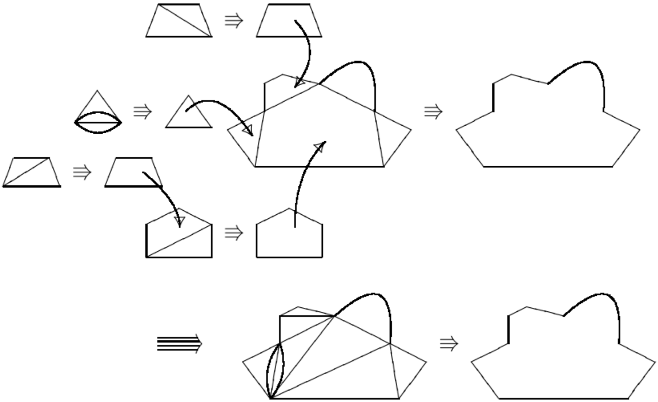


Pasting schemes

However...

Pasting schemes

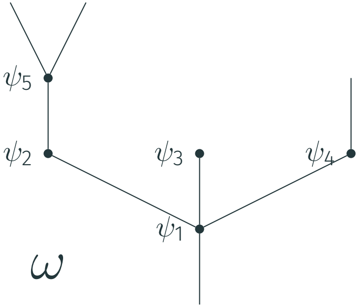
However...



(this is representing an opetope of dimension only 4, diagram from [CL])

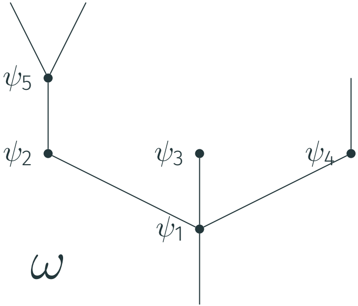
Trees (naive)

What's wrong with trees as before?



Trees (naive)

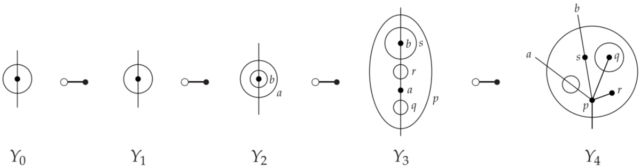
What's wrong with trees as before?



Each node is decorated by an $(n-1)$-opetope (ψ_1, ...), so it should come with its own tree, and likewise for the nodes of that tree, etc... It quickly gets out of hand.

This tree approach has been reworked and improved by Batanin, Joyal, Kock, and Mascari [KJBM10].

A *constellation* is a sequence of trees with circles satisfying some conditions.



Syntax

Addresses

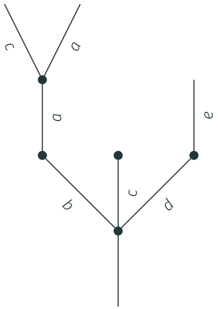
We seek to make the notion of occurrences more precise.

Addresses

We seek to make the notion of occurrences more precise.

Observation

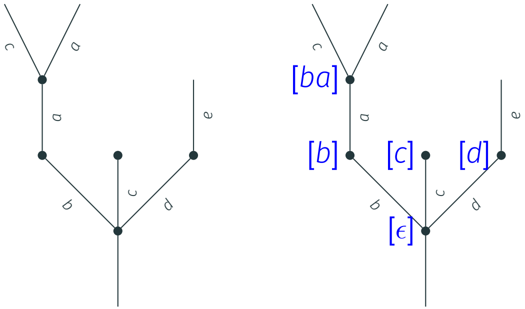
If each edge has a name that is unique among its siblings, then each node can be uniquely identified by a *path*. In turn, the corolla associated to the tree has names for its input edges.



We seek to make the notion of occurrences more precise.

Observation

If each edge has a name that is unique among its siblings, then each node can be uniquely identified by a *path*. In turn, the corolla associated to the tree has names for its input edges.

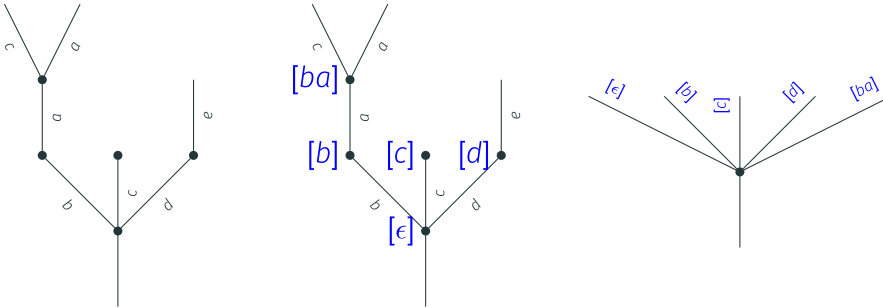


Addresses

We seek to make the notion of occurrences more precise.

Observation

If each edge has a name that is unique among its siblings, then each node can be uniquely identified by a *path*. In turn, the corolla associated to the tree has names for its input edges.



Addresses: base case

In order for all opetopes (of dimension ≥ 2) to have such addresses, we need to bootstrap the process in dimension 1:

Addresses: base case

In order for all opetopes (of dimension ≥ 2) to have such addresses, we need to bootstrap the process in dimension 1:

We decide that the name of the input edge is $*$ (in blue).

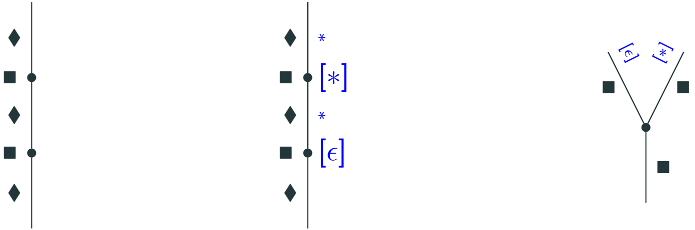
Addresses: dimension 2

As an example of how the process continues:

Addresses: dimension 2

As an example of how the process continues:

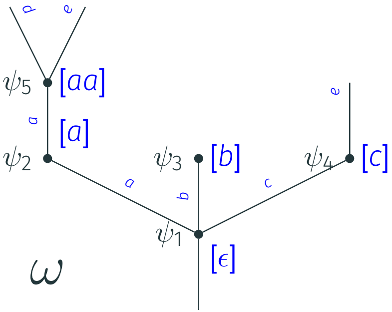
As an example of how the process continues:



Idea

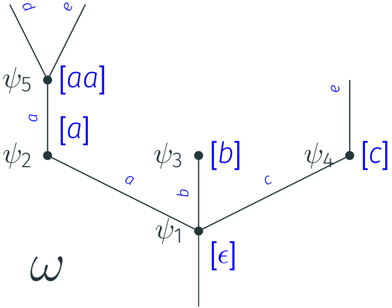
Our take on the representation of opetopes is a syntactic one.

Take ω an n-opetope, for $n \geq 2$, as below:



Idea

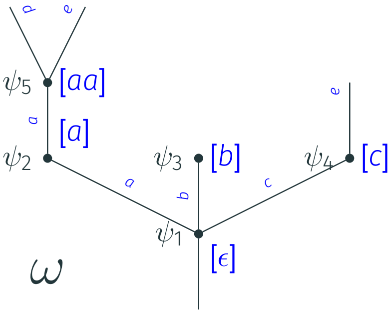
Our take on the representation of opetopes is a syntactic one.
Take ω an n-opetope, for $n \geq 2$, as below:



Observation

In ω, each node is decorated by a $(n-1)$-opetope

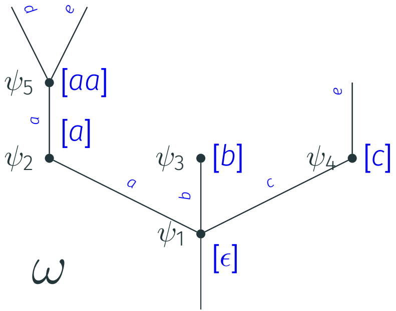
Our take on the representation of opetopes is a syntactic one. Take ω an n-opetope, for $n \geq 2$, as below:



Observation

In ω, each node is decorated by a $(n-1)$-opetope, and each node has an address.

Thus, the syntactic representation (or *coding*) $\ulcorner\omega\urcorner$ of ω is



$$\begin{cases} [\epsilon] \leftarrow \ulcorner\psi_1\urcorner \\ [a] \leftarrow \ulcorner\psi_2\urcorner \\ [b] \leftarrow \ulcorner\psi_3\urcorner \\ [c] \leftarrow \ulcorner\psi_4\urcorner \\ [aa] \leftarrow \ulcorner\psi_5\urcorner \end{cases}$$

Idea

The 0 and 1-opetopes are coded by

$$\ulcorner \blacklozenge \urcorner = \blacklozenge, \qquad \ulcorner \blacksquare \urcorner = \blacksquare = \left\{ * \leftarrow \blacklozenge \right. .$$

Idea

The 0 and 1-opetopes are coded by

$$\ulcorner \blacklozenge \urcorner = \blacklozenge, \qquad \ulcorner \blacksquare \urcorner = \blacksquare = \left\{ * \leftarrow \blacklozenge \right. .$$

Degenerate opetopes are coded as follows:

$$\phi \Bigg| \qquad\qquad\qquad \left\{ \left\{ \ulcorner \phi \urcorner \right. \right.$$

$$
\begin{array}{rcll}
0\texttt{-addr} & ::= & * & \\
n\texttt{-addr} & ::= & [\epsilon] & n \geq 1 \\
 & | & [(n-1)\texttt{addr} \; \cdots \; (n-1)\texttt{addr}] & n \geq 1 \\
0\texttt{-preopt} & ::= & \blacklozenge & \\
n\texttt{-preopt} & ::= & \begin{cases} (n-1)\texttt{-addr} \leftarrow (n-1)\texttt{-preopt} \\ \vdots \\ (n-1)\texttt{-addr} \leftarrow (n-1)\texttt{-preopt} \end{cases} & n \geq 1 \\
 & | & \Big\{\Big\{ (n-2)\texttt{-preopt} & n \geq 2
\end{array}
$$

Preopetopes

Example

$$\left\{\begin{array}{l} [\epsilon] \leftarrow \left\{\begin{array}{l} [\epsilon] \leftarrow \left\{\begin{array}{l} [\epsilon] \leftarrow \left\{ * \leftarrow \blacklozenge \right. \\ [*] \leftarrow \left\{ * \leftarrow \blacklozenge \right. \end{array}\right. \\ [[*]] \leftarrow \left\{ [\epsilon] \leftarrow \left\{ * \leftarrow \blacklozenge \right.\right. \end{array}\right. \\ [[[*]]] \leftarrow \left\{\left\{\left\{ * \leftarrow \blacklozenge \right.\right.\right. \end{array}\right.$$

Preopetopes

We thus define a coding function

$$\ulcorner - \urcorner : n\texttt{-opetopes} \longrightarrow n\texttt{-preopt}.$$

Observation

Clearly, if $\omega \neq \omega'$, then $\ulcorner \omega \urcorner \neq \ulcorner \omega' \urcorner$.

Preopetopes

We thus define a coding function

$$\ulcorner - \urcorner : n\texttt{-opetopes} \longrightarrow n\texttt{-preopt}.$$

Observation

Clearly, if $\omega \neq \omega'$, then $\ulcorner \omega \urcorner \neq \ulcorner \omega' \urcorner$.

- If they have the same dimension, then their tree structure differ and the result is immediate.

We thus define a coding function

$$\ulcorner - \urcorner : n\texttt{-opetopes} \longrightarrow n\texttt{-preopt}.$$

Observation

Clearly, if $\omega \neq \omega'$, then $\ulcorner \omega \urcorner \neq \ulcorner \omega' \urcorner$.

- If they have the same dimension, then their tree structure differ and the result is immediate.
- The "depth" of $\ulcorner \omega \urcorner$ is the dimension of ω, so if ω and ω' don't have the same dimension, then $\ulcorner \omega \urcorner$ and $\ulcorner \omega' \urcorner$ don't have the same depth, hence are different.

Question

Is $\ulcorner - \urcorner$ surjective? In other words, is all preopetope the coding of an actual opetope?

Question

Is $\ulcorner - \urcorner$ surjective? In other words, is all preopetope the coding of an actual opetope?

No!

$$\left\{ [*] \leftarrow \ulcorner \psi \urcorner \right.$$

Question

Is $\ulcorner - \urcorner$ surjective? In other words, is all preopetope the coding of an actual opetope?

No!

$$\left\{ [*] \leftarrow \ulcorner \psi \urcorner \right.$$

Goal

Create a deduction system where derivable preopetopes are exactly the codings of opetopes.

Inference rules: structure of a sequent

Our system will deal with sequents of the form

$$\Gamma \vdash \mathfrak{p}$$

where

- $\mathfrak{p}$ is a preopetope;
- Γ is the *set* of the (addresses of the) *leaves* of $\mathfrak{p}$.

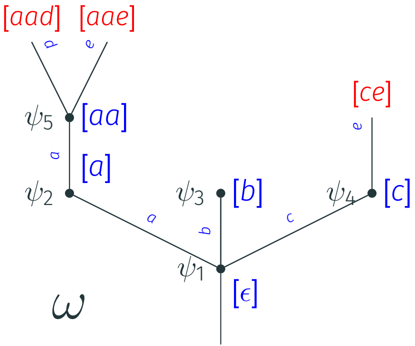
Inference rules: structure of a sequent

Our system will deal with sequents of the form

$$\Gamma \vdash \mathfrak{p}$$

where

- $\mathfrak{p}$ is a preopetope;
- Γ is the *set* of the (addresses of the) *leaves* of $\mathfrak{p}$.



$$\Gamma = \{[aad], [aae], [ce]\}$$

Inference rules: `point`

In order to bootstrap the derivation process, rule `point` creates a 0-opetope $\blacklozenge$ from no premise.

$$\frac{}{\vdash \blacklozenge}\ \texttt{point}$$

Inference rules: degen

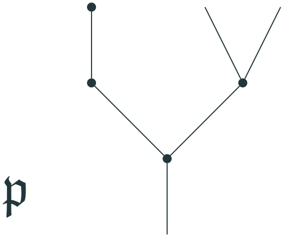
Given a n-preopetope $\mathfrak{p}$, we can create the *degenerate* $(n+2)$-*preopetope* at $\mathfrak{p}$, i.e. a tree with no node, and whose unique edge is decorated by $\mathfrak{p}$.

$$\frac{\Gamma \vdash \mathfrak{p}}{\vdash \{\{\mathfrak{p}} \ \mathsf{degen}$$

Inference rules: degen

Given a n-preopetope $\mathfrak{p}$, we can create the *degenerate* $(n+2)$-*preopetope* at $\mathfrak{p}$, i.e. a tree with no node, and whose unique edge is decorated by $\mathfrak{p}$.

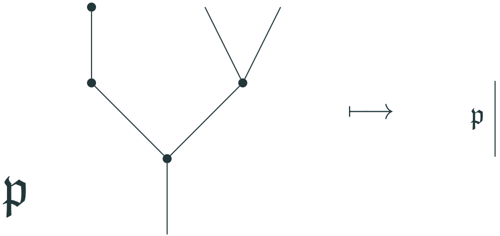
$$\frac{\Gamma \vdash \mathfrak{p}}{\vdash \{\{\mathfrak{p}} \text{ degen}$$



Inference rules: degen

Given a n-preopetope $\mathfrak{p}$, we can create the *degenerate* $(n+2)$*-preopetope* at $\mathfrak{p}$, i.e. a tree with no node, and whose unique edge is decorated by $\mathfrak{p}$.

$$\frac{\Gamma \vdash \mathfrak{p}}{\vdash \left\{ \left\{ \mathfrak{p} \right.\right.} \,\mathsf{degen}$$



Inference rules: `shift`

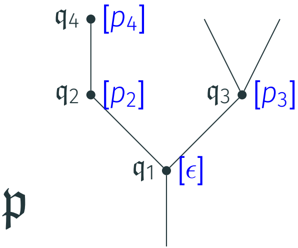
Given a n-preopetope $\mathfrak{p} = \begin{cases} [p_1] \leftarrow \mathfrak{q}_1 \\ \vdots \\ [p_k] \leftarrow \mathfrak{q}_k \end{cases}$, we can create an $(n+1)$-preopetope, the corolla whose input edges correspond to the nodes of $\mathfrak{p}$.

$$\frac{\Gamma \vdash \mathfrak{p}}{[[p_1]], \ldots, [[p_k]] \vdash \left\{ [\epsilon] \leftarrow \mathfrak{p} \right.} \,\texttt{shift}$$

Inference rules: `shift`

Given a n-preopetope $\mathfrak{p} = \begin{cases} [p_1] & \leftarrow & \mathfrak{q}_1 \\ & \vdots & \\ [p_k] & \leftarrow & \mathfrak{q}_k \end{cases}$, we can create an $(n+1)$-preopetope, the corolla whose input edges correspond to the nodes of $\mathfrak{p}$.

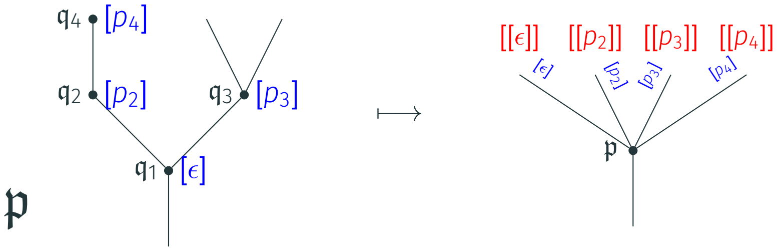
$$\frac{\Gamma \vdash \mathfrak{p}}{[[p_1]], \ldots, [[p_k]] \vdash \left\{ [\epsilon] \leftarrow \mathfrak{p} \right.} \; \texttt{shift}$$



Inference rules: shift

Given a n-preopetope $\mathfrak{p} = \begin{cases} [p_1] & \leftarrow & \mathfrak{q}_1 \\ & \vdots & \\ [p_k] & \leftarrow & \mathfrak{q}_k \end{cases}$, we can create an $(n+1)$-preopetope, the corolla whose input edges correspond to the nodes of $\mathfrak{p}$.

$$\frac{\Gamma \vdash \mathfrak{p}}{[[p_1]], \ldots, [[p_k]] \vdash \left\{ [\epsilon] \leftarrow \mathfrak{p} \right.} \texttt{shift}$$



Inference rules: `graft`

We assume the existence of a magical *target* function

$$\mathsf{t} : \{n\text{-preopt.}\} \longrightarrow \{(n-1)\text{-preopt.}\}$$

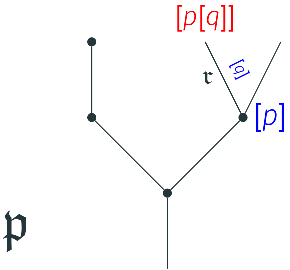
Inference rules: `graft`

Given an n-preopetope $\mathfrak{p}$ on the left with a leaf at address $[p[q]]$, and an $(n-1)$-preopetope $\mathfrak{q}$, with target $\mathsf{t}\,\mathfrak{q} = \mathfrak{r}$, we may graft $\mathfrak{q}$ on $\mathfrak{p}$.

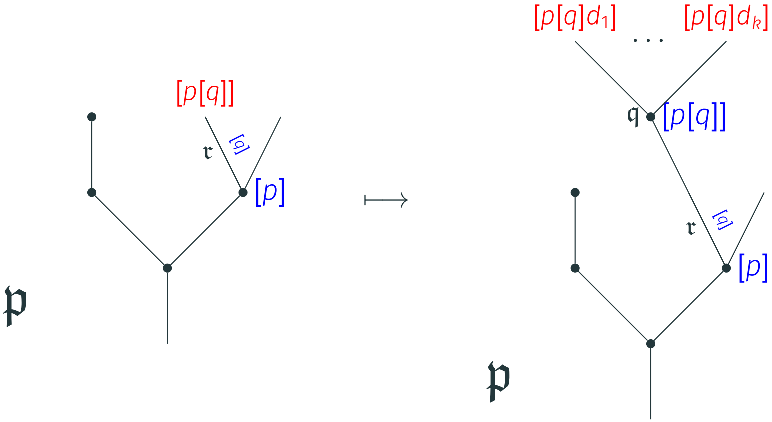
$$\frac{\Gamma, [p[q]] \vdash \left\{ \begin{array}{l} \quad\vdots \\ [p] \leftarrow \left\{ \begin{array}{l} \quad\vdots \\ [q] \leftarrow \mathfrak{r} \\ \quad\vdots \end{array}\right. \\ \quad\vdots \end{array}\right. \qquad \Delta \vdash \mathfrak{q}}{\Gamma, \Xi \vdash \left\{ \begin{array}{l} \quad\vdots \\ [p] \leftarrow \left\{ \begin{array}{l} \quad\vdots \\ [q] \leftarrow \mathfrak{r} \\ \quad\vdots \end{array}\right. \\ \quad\vdots \\ [p[q]] \leftarrow \mathfrak{q} \end{array}\right.}\ \texttt{graft}$$

where $\Xi = \{[p[q]d] \mid [d] \text{ address in } \mathfrak{q}\}$.

Inference rules: `graft`



Inference rules: `graft`



Equivalence theorem

Theorem (Curien–HT–Mimram)
The set of derivable preopetopes is in one-to-one correspondence with the set of Leinster opetopes [Lei, KJBM10].

Theorem (Curien–HT–Mimram)
The set of derivable preopetopes is in one-to-one correspondence with the set of Leinster opetopes [Lei, KJBM10].

The proof essentially boils down to make the previous drawings rigorous.

We now refer to derivable preopetopes as simply opetopes.

Well-formation algorithm

Theorem (Curien–HT–Mimram)
Opetopes form a decidable subset of the set of preopetopes.

Well-formation algorithm

Theorem (Curien–HT–Mimram)
Opetopes form a decidable subset of the set of preopetopes.

The naive algorithm, that seeks to deconstruct the potential proof tree of a given preopetope, runs in exponential time w.r.t. the number of preopetopes involved in each dimension.

Conclusion

To summarize

- Opetopes are tricky, highly inductive objects.
- Following up on major advancements on the matter [Lei, KJBM10], we propose a purely syntactic definition.
- We believe such explicit “writable” expressions will make opetopes easier to use for subsequent work.

Thank you for your attention!

References i

Eugenia Cheng and Aaron Lauda.
Higher-Dimensional Categories: an illustrated guide book.

Joachim Kock, André Joyal, Michael Batanin, and Jean-François Mascari.
Polynomial functors and opetopes.
Advances in Mathematics, 224(6):2690–2737, 2010.

Tom Leinster.
Higher operads, higher categories.