

Single-set cubical categories and their formalisation with a proof assistant

Philippe Malbos¹, Tanguy Massacrier¹, Georg Struth²

¹Université Claude Bernard Lyon 1

²University of Sheffield

<https://arxiv.org/abs/2401.10553>

<https://www.isa-afp.org/entries/CubicalCategories.html>

Single-set cubical categories

- Cubical sets and categories are a categorical description of cubes, their faces and their compositions. They provide a language used for:
 - studying homotopy,
 - studying higher-dimensional rewriting,
 - modelling concurrency using higher dimensional automata,
 - modelling homotopy type theory.
- We want to study the process of formalisation. We present an alternative, single-set, description of cubical categories.
- A single-set approach is easier to formalise and to compute with. We use the proof assistant **Isabelle** to formalise these categories and to help us find their definition.

The geometry of computations is cubical

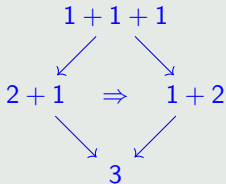
We want to study computations in an algebraic structure using rewriting.

Example: expressions in $(\mathbb{N}, +)$.

We replace equalities by oriented arrows:

$$1 + 1 \longrightarrow 2$$

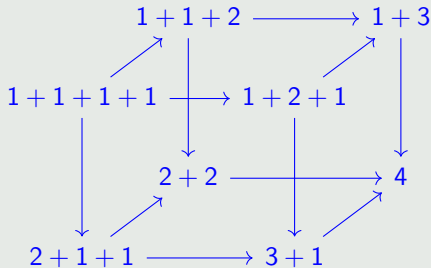
Compare relations:



The geometry of computations is cubical

Example: expressions in $(\mathbb{N}, +)$.

Compare relations between relations:



- Cubical categories provide a language for formalising computations.
- We apply it to abstract rewriting systems in a forthcoming work.

Cubical categories

Definition (Brown, Higgins, 1981):

- A **cubical ω -category** is the data of:
 - a set \mathcal{C}_k of k -cells for $k \in \mathbb{N}$,
 - face maps $\partial_{k,i}^\alpha : \mathcal{C}_k \rightarrow \mathcal{C}_{k-1}$ for $1 \leq i \leq k$ and $\alpha \in \{-, +\}$,
 - degeneracies $\epsilon_{k,i} : \mathcal{C}_{k-1} \rightarrow \mathcal{C}_k$ for $1 \leq i \leq k$,
 - compositions $\star_{k,i} : \mathcal{C}_k \times_{k,i} \mathcal{C}_k \rightarrow \mathcal{C}_k$ for $1 \leq i \leq k$,

satisfying some compatibility conditions.

$$\mathcal{C}_0 \begin{array}{c} \xleftarrow{\partial_{1,1}^-} \\ \xleftarrow{\partial_{1,1}^+} \end{array} \mathcal{C}_1 \begin{array}{c} \xleftarrow{\partial_{2,1}^-} \\ \xleftarrow{\partial_{2,2}^+} \end{array} \mathcal{C}_2 \quad \dots \quad \mathcal{C}_{k-1} \begin{array}{c} \xleftarrow{\partial_{k,1}^-} \\ \xleftarrow{\partial_{k,k}^+} \end{array} \mathcal{C}_k \quad \dots$$

- A cubical n -category is the same but we forget the structure after dimension n .

Cells and dimensions

The shapes of cells are as follows:

- 0-cells are points,

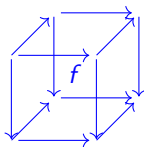
$$x \xrightarrow{f} y$$

- 1-cells,



- 2-cells,

- 3-cells,



Face maps

In every direction i , there are two faces: $\partial_{k,i}^- = \partial_i^-$ (source) and $\partial_{k,i}^+ = \partial_i^+$ (target).



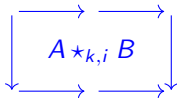
$$\begin{array}{ccc}
 \partial_i^- \partial_j^- x & \xrightarrow{\partial_i^- x} & \partial_i^- \partial_j^+ x \\
 \partial_j^- x \downarrow & x & \downarrow \partial_j^+ x \\
 \partial_i^+ \partial_j^- x & \xrightarrow{\partial_i^+ x} & \partial_i^+ \partial_j^+ x
 \end{array}$$

Compositions

When faces $\partial_{k,i}^+ A = \partial_{k,i}^- B$ of two k -cells coincide



then we can $\star_{k,i}$ -compose them by 'glueing' them along direction i .



Degeneracies

From a 0-cell x we get an identity 1-cell.

$$\bullet^x \xrightarrow{\epsilon_{1,1}} x = x$$

From a 1-cell we get two degenerate 2-cells.

$$\begin{array}{ccc}
 & & \begin{array}{ccc} x & \xrightarrow{f} & y \\ \parallel & & \parallel \\ x & \xrightarrow{f} & y \end{array} \\
 & \nearrow \epsilon_{2,1} & \\
 x \xrightarrow{f} y & & \\
 & \searrow \epsilon_{2,2} & \\
 & & \begin{array}{ccc} x & = & x \\ f \downarrow & & \downarrow f \\ y & = & y \end{array}
 \end{array}$$

The degeneracies $\epsilon_{k,i}$ are the identities for the $\star_{k,i}$ -composition.

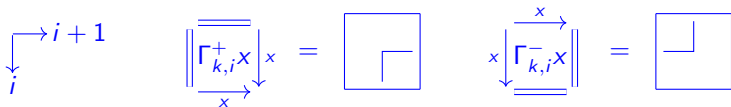
The categories \mathbf{Cub}_n

A functor $F : \mathcal{C} \rightarrow \mathcal{D}$ is a family of maps $F_k : \mathcal{C}_k \rightarrow \mathcal{D}_k$ preserving the structure (face maps, degeneracies, compositions).

This defines the categories \mathbf{Cub}_n for $n \in \mathbb{N} \cup \{\omega\}$.

Connections and equivalence with globular categories

Connections are 'twisted' degeneracies. They bend the wires between different directions.



By adding functors, this defines the categories Cub_n^{Γ} for $n \in \mathbb{N} \cup \{\omega\}$.

Theorem (Al-Agl, Brown, Steiner, 2001)

For $n \in \mathbb{N} \cup \{\omega\}$, $\text{Cub}_n^{\Gamma} \simeq \text{Cat}_n$.

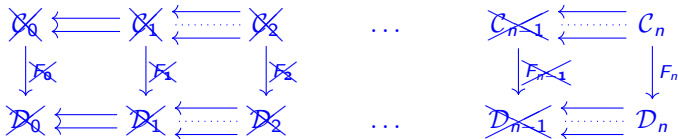
From classical to single-set

- Alternative model of cubical categories.
- Idea: the low dimensional cells are already encoded in higher dimensions as identity cells, the degeneracies.



From classical to single-set

- Easier to compute with, because we only have one set containing all the cells. We don't treat the cells of different dimensions separately, but all at once.
 - For instance:* a functor is only a function $F : \mathcal{S} \rightarrow \mathcal{T}$ respecting face maps, symmetries and compositions.



- Easier to formalise in a proof assistant, because the dimension of cells is not captured with types but functionally through fixed-point properties.
- We formalised single-set cubical categories in Isabelle. Isabelle used to develop their axiomatisation.

Dimension 1

Definition (MacLane, 1971):

A **single-set category** \mathcal{S} is the data of:

- a set \mathcal{S} of cells,
- face maps $\delta^- : \mathcal{S} \rightarrow \mathcal{S}$ (source) and $\delta^+ : \mathcal{S} \rightarrow \mathcal{S}$ (target),
- partially defined composition map \otimes from $\mathcal{S} \times \mathcal{S}$ to \mathcal{S} ,

where $x \otimes y$ is defined if and only if $\delta^+ x = \delta^- y$, in which case

$$\delta^-(x \otimes y) = \delta^- x, \qquad \delta^+(x \otimes y) = \delta^+ y,$$

$$x \otimes \delta^+ x = x, \qquad \delta^- x \otimes x = x,$$

$$x \otimes (y \otimes z) = x \otimes (y \otimes z)$$

$$\delta^- \delta^- x = \delta^- x = \delta^+ \delta^- x,$$

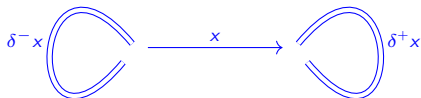
$$\delta^+ \delta^+ x = \delta^+ x = \delta^- \delta^+ x.$$

Dimension 1

The fixed points of the face maps

$$\mathcal{S}^\delta = \{x \in \mathcal{S} \mid \delta^-x = x\} = \{x \in \mathcal{S} \mid \delta^+x = x\}$$

are the identity arrows for the composition, and they correspond to the 0-cells in classical categories.



Single-set cubical categories

Definition:

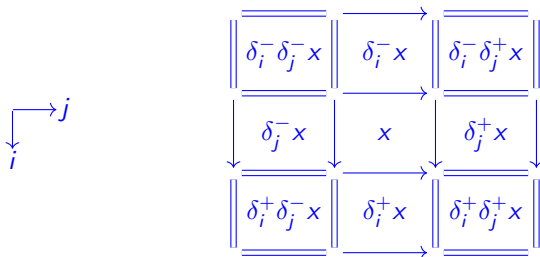
- A **single-set cubical ω -category** \mathcal{S} is the data of:
 - a family of single-set categories $(\mathcal{S}, \delta_i^-, \delta_i^+, \otimes_i)_{i \geq 1}$,
 - symmetries $s_i : \mathcal{S} \rightarrow \mathcal{S}$ for $i \geq 1$,
 - reverse symmetries $\tilde{s}_i : \mathcal{S} \rightarrow \mathcal{S}$ for $i \geq 1$,

satisfying some compatibility conditions.

- A single-set cubical n -category is the same but we forget the structure after dimension n .

Faces

The faces are themselves cells:



Symmetries

Let's recall that every 1-cell can be seen as a degenerate 2-cell in two ways.

$$\begin{array}{ccc} x & \xrightarrow{f} & y \\ \parallel & & \parallel \\ x & \xrightarrow{f} & y \end{array}$$

$$\begin{array}{ccc} x & \equiv & x \\ f \downarrow & & \downarrow f \\ y & \equiv & y \end{array}$$

We need a way to identify them: symmetries.

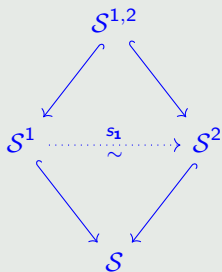
The symmetries exchange directions: s_i sends identities for \otimes_i -composition to identities for \otimes_{i+1} -composition.

Lattice of fixed points

How do we recover low-dimensional cells?

Define $\mathcal{S}^i = \{x \in \mathcal{S} \mid \delta_i^\pm x = x\}$, the set of fixed points for the face maps in direction i , and $\mathcal{S}^I = \bigcap_{i \in I} \mathcal{S}^i$. We get inclusions forming a lattice.

Example in dimension 2.



} 0-cells



} 1-cells

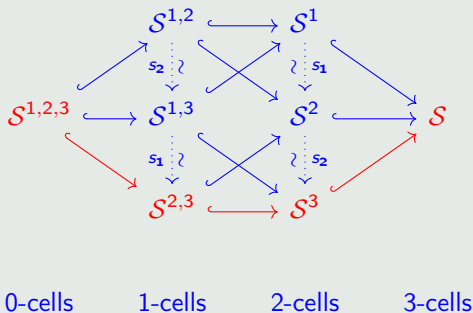


} 2-cells



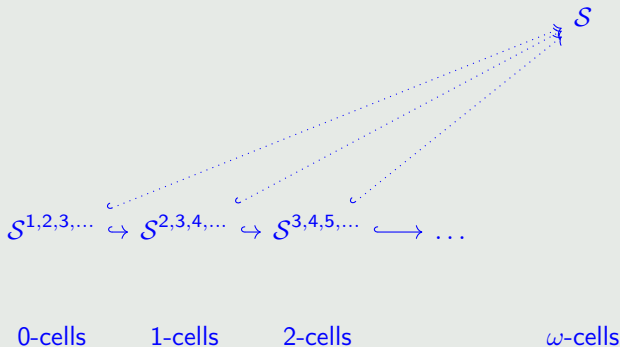
Lattice of fixed points

Example in dimension 3.



Lattice of fixed points

Example in dimension ω .



The categories \mathbf{SCub}_n and \mathbf{SCub}_n^γ

- A functor $F : \mathcal{S} \rightarrow \mathcal{T}$ is a map preserving the structure.
This defines the categories \mathbf{SCub}_n for $n \in \mathbb{N} \cup \{\omega\}$.
- As before, we can add connections to the structure: maps $\gamma_i^\alpha : \mathcal{S} \rightarrow \mathcal{S}$ for $i \geq 1$ and $\alpha \in \{-, +\}$ satisfying some conditions.
Adding functors, we get categories \mathbf{SCub}_n^γ for $n \in \mathbb{N} \cup \{\omega\}$.

Equivalence with classical cubical categories

- We can recover the low-dimensional structure using the fixed points of the face maps, hence:

Theorem

For $n \in \mathbb{N} \cup \{\omega\}$, $\text{SCub}_n \simeq \text{Cub}_n$ and $\text{SCub}_n^\gamma \simeq \text{Cub}_n^\Gamma$.

- Isabelle was used to find the definition of single-set cubical categories and to prove the above equivalence.

Experimental mathematics with Isabelle

How did we find the definition of single-set cubical categories?

- We tried to copy the classical cubical axioms. Example:

$$\partial_{k,i+1}^\alpha \Gamma_{k,i}^\alpha x = x \quad \text{for } x \text{ in } \mathcal{C}_{k-1} \quad \rightsquigarrow \quad \delta_{i+1}^\alpha \gamma_i^\alpha x = \underset{\uparrow}{s_i} x \quad \text{for } x \text{ in } \mathcal{S}^i$$

symmetries had to be introduced

- We added all the axioms needed to show the equivalence of categories.
- We used Isabelle automated proof search tools to show the redundancy of some axioms, and removed them from the definition. Example:

$$\partial_{k,i}^\alpha \Gamma_{k,i}^{-\alpha} = \epsilon_{k-1,i} \partial_{k-1,i}^\alpha \quad \rightsquigarrow \quad \text{not needed in single-set}$$

- It simplified the proof of the equivalence.

Conclusion and perspectives

- We introduced a single-set axiomatisation of cubical categories.
- We showed they are equivalent to classical cubical categories.
- We implemented their definition in Isabelle.
- We can require our cells to be invertible. This gives the categories $\text{Cub}_{(n,p)}^\Gamma$ and $\text{SCub}_{(n,p)}^\gamma$, to which we extended our results.
- These (n, p) -categories are used in rewriting. Indeed an (ω, p) -category \mathcal{C} presents the p -category obtained by quotienting by the equivalence generated by the $(p + 1)$ -cells.

$$x \longrightarrow \longleftarrow \longrightarrow \quad \dots \quad \longrightarrow y$$

Finding such presentations by free well-behaved (ω, p) -categories, called resolutions, is a goal of higher-dimensional rewriting. In a forthcoming work we study the case $p = 0$, that is abstract rewriting systems.

Conclusion and perspectives

Next we want to study higher-dimensional rewriting properties in the single-set cubical setting:

- normalisation strategies, which are deterministic choices of reduction paths from one cell to another reduced one,
- proofs of Church-Rosser theorem and Newman's lemma, which characterise confluence properties of rewriting systems,
- polygraphic resolutions of higher categories.

Thank you.

Appendix 1: single-set cubical ω -category

A **single-set cubical ω -category** consists of a family of single-set categories $(\mathcal{S}, \delta_i^-, \delta_i^+, \otimes_i)_{i \in \mathbb{N}_+}$ with *symmetry maps* $s_i : \mathcal{S} \rightarrow \mathcal{S}$ and *reverse symmetry maps* $\tilde{s}_i : \mathcal{S} \rightarrow \mathcal{S}$ for each $i \in \mathbb{N}_+$. These satisfy, for all $w, x, y, z \in \mathcal{S}$ and $i, j \in \mathbb{N}_+$,

- (i) $\delta_i^\alpha \delta_j^\beta = \delta_j^\beta \delta_i^\alpha$ if $i \neq j$,
- (ii) $\delta_i^\alpha(x \otimes_j y) = \delta_i^\alpha x \otimes_j \delta_i^\alpha y$ if $i \neq j$ and $\Delta_j(x, y)$,
- (iii) $(w \otimes_i x) \otimes_j (y \otimes_i z) = (w \otimes_j y) \otimes_i (x \otimes_j z)$ if $i \neq j$, $\Delta_i(w, x)$, $\Delta_i(y, z)$, $\Delta_j(w, y)$ and $\Delta_j(x, z)$,
- (iv) $s_i(\mathcal{S}^i) \subseteq \mathcal{S}^{i+1}$ and $\tilde{s}_i(\mathcal{S}^{i+1}) \subseteq \mathcal{S}^i$,
- (v) $\tilde{s}_i s_i x = x$ and $s_i \tilde{s}_i y = y$ if $x \in \mathcal{S}^i$ and $y \in \mathcal{S}^{i+1}$,
- (vi) $\delta_j^\alpha s_j x = s_j \delta_{j+1}^\alpha x$ and $\delta_i^\alpha s_j x = s_j \delta_i^\alpha x$ if $i \neq j, j+1$ and $x \in \mathcal{S}^j$,
- (vii) $s_i(x \otimes_{i+1} y) = s_i x \otimes_i s_i y$ and $s_i(x \otimes_j y) = s_i x \otimes_j s_i y$ if $j \neq i, i+1$, $x, y \in \mathcal{S}^i$ and $\Delta_j(x, y)$,
- (viii) $s_i x = x$ if $x \in \mathcal{S}^i \cap \mathcal{S}^{i+1}$,
- (ix) $s_i s_j x = s_j s_i x$ if $|i - j| \geq 2$ and $x \in \mathcal{S}^i \cap \mathcal{S}^j$,
- (x) $\exists k \in \mathbb{N} \forall i \geq k+1, x \in \mathcal{S}^i$.

Appendix 2: with connections

A **single-set cubical ω -category with connections** is a single-set cubical ω -category \mathcal{S} with *connection maps* $\gamma_i^\alpha : \mathcal{S} \rightarrow \mathcal{S}$, for all $i \in \mathbb{N}_+$ and $\alpha \in \{-, +\}$. These satisfy, for all $i, j \in \mathbb{N}_+$,

$$(i) \quad \delta_j^\alpha \gamma_j^\alpha x = x, \quad \delta_{j+1}^\alpha \gamma_j^\alpha x = s_j x \quad \text{and} \quad \delta_i^\alpha \gamma_j^\beta x = \gamma_j^\beta \delta_i^\alpha x \quad \text{if } i \neq j, j+1 \text{ and } x \in \mathcal{S}^j,$$

(ii) if $j \neq i, i+1$ and $x, y \in \mathcal{S}^i$, then

$$\Delta_{i+1}(x, y) \Rightarrow \gamma_i^+(x \otimes_{i+1} y) = (\gamma_i^+ x \otimes_{i+1} s_i x) \otimes_i (x \otimes_{i+1} \gamma_i^+ y),$$

$$\Delta_{i+1}(x, y) \Rightarrow \gamma_i^-(x \otimes_{i+1} y) = (\gamma_i^- x \otimes_{i+1} y) \otimes_i (s_i y \otimes_{i+1} \gamma_i^- y),$$

$$\Delta_j(x, y) \Rightarrow \gamma_i^\alpha(x \otimes_j y) = \gamma_i^\alpha x \otimes_j \gamma_i^\alpha y,$$

$$(iii) \quad \gamma_i^\alpha x = x \quad \text{if } x \in \mathcal{S}^{i, i+1},$$

$$(iv) \quad \gamma_i^+ x \otimes_{i+1} \gamma_i^- x = x \quad \text{and} \quad \gamma_i^+ x \otimes_i \gamma_i^- x = s_i x \quad \text{if } x \in \mathcal{S}^i,$$

$$(v) \quad \gamma_i^\alpha \gamma_j^\beta x = \gamma_j^\beta \gamma_i^\alpha x \quad \text{if } |i-j| \geq 2 \text{ and } x \in \mathcal{S}^{i, j},$$

$$(vi) \quad s_{i+1} s_i \gamma_{i+1}^\alpha x = \gamma_i^\alpha s_{i+1} x \quad \text{if } x \in \mathcal{S}^{i, i+1}.$$

Appendix 3: Isabelle

File Edit Search Markers Folding View Utilities Macros Plugins Help

CubicalCategories.thy (~\Dossiers Tanguy\Cours\Thèse\Projets\Projets Git\lab\isabelle)

```

subsection <Semi-cubical  $\omega$ -categories>

text <We first define a class for cubical  $\omega$ -categories without symmetries.>

class semi_cubical_omega_category = icategory +
  assumes face_comm: " $i \neq j \implies \partial i \alpha \circ \partial j \beta = \partial j \beta \circ \partial i \alpha$ "
  and face_func: " $i \neq j \implies \text{DD } j \ x \ y \implies \partial i \alpha \ (x \otimes_{\omega} j \ y) = \partial i \alpha \ x \ \otimes_{\omega} j \ \partial i \alpha \ y$ "
  and interchange: " $i \neq j \implies \text{DD } i \ w \ x \implies \text{DD } i \ y \ z \implies \text{DD } j \ w \ y \implies \text{DD } j \ x \ z$   

 $\implies (w \ \otimes_{\omega} i \ x) \ \otimes_{\omega} j \ (y \ \otimes_{\omega} i \ z) = (w \ \otimes_{\omega} j \ y) \ \otimes_{\omega} i \ (x \ \otimes_{\omega} j \ z)$ "
  and fin_fix: " $\exists k. \forall i. k \leq i \implies \text{fF } x \ i \ x$ "

begin

lemma fin_fix_var: "fin_dim x"
  by (simp add: local.fin_fix)

lemma pcomp_face_func_DD: " $i \neq j \implies \text{DD } j \ x \ y \implies \text{DD } j \ (\partial i \alpha \ x) \ (\partial i \alpha \ y)$ "
  by (metis comp_apply icat.st_local local.face_comm)

lemma comp_face_func: " $i \neq j \implies (\partial \partial i \alpha) \ (x \ \otimes_{\omega} j \ y) \subseteq \partial i \alpha \ x \ \otimes_{\omega} j \ \partial i \alpha \ y$ "
  
```

Provers: cvc4 verit z3 e spass vampire zipperposition Isar proofs Try methods Apply 00%

Output: Query Sledgehammer Symbols

(isabelle,isabelle.UTF-8-Isabelle) | nmro UG 05 164/512MB 10:19 PM

Appendix 3: Isabelle

The screenshot shows the Isabelle proof assistant interface. The main window displays a theorem script for defining cubical categories. The script includes several axioms and a lemma. The interface also shows a file browser on the left, a prover status on the right, and a prover selection menu at the bottom.

```

text <Next we define a class for cubical  $\omega$ -categories.>

class cubical_omega_category = semi_cubical_omega_category + symmetry_ops +
  assumes sym_type: " $\sigma\ i\ (\text{face\_fix } i) \subseteq \text{face\_fix } (i + 1)$ "
  and inv_sym_type: " $\partial\ i\ (\text{face\_fix } (i + 1)) \subseteq \text{face\_fix } i$ "
  and sym_inv_sym: " $\text{fFx } (i + 1)\ x \implies \sigma\ i\ (\partial\ i\ x) = x$ "
  and inv_sym_sym: " $\text{fFx } i\ x \implies \partial\ i\ (\sigma\ i\ x) = x$ "
  and sym_facel: " $\text{fFx } i\ x \implies \partial\ i\ \alpha\ (\sigma\ i\ x) = \sigma\ i\ (\partial\ (i + 1)\ \alpha\ x)$ "
  and sym_face2: " $i \neq j \implies i \neq j + 1 \implies \text{fFx } j\ x \implies \partial\ i\ \alpha\ (\sigma\ j\ x) = \sigma\ j\ (\partial\ i\ \alpha\ x)$ "
  and sym_func: " $i \neq j \implies \text{fFx } i\ x \implies \text{fFx } i\ y \implies \text{DD } j\ x\ y \implies$ 
     $\sigma\ i\ (x \otimes_{j\ \sigma}\ y) = (\text{if } j = i + 1 \text{ then } \sigma\ i\ x \otimes_{i\ \sigma}\ \sigma\ i\ y \text{ else } \sigma\ i\ x \otimes_{j\ \sigma}\ \sigma\ i\ y)$ "
  and sym_fix: " $\text{fFx } i\ x \implies \text{fFx } (i + 1)\ x \implies \sigma\ i\ x = x$ "
  and sym_sym_braid: " $\text{diffSup } i\ j\ 2 \implies \text{fFx } i\ x \implies \text{fFx } j\ x \implies \sigma\ i\ (\sigma\ j\ x) = \sigma\ j\ (\sigma\ i\ x)$ "

begin

text <First we prove variants of the axioms.>

lemma sym_type_var: " $\text{fFx } i\ x \implies \text{fFx } (i + 1)\ (\sigma\ i\ x)$ "
  by (meson image_subset_iff local.face_fix_prop local.sym_type)
  
```

Provers: cvc4 vent 23 e spass vampire zipperposition Isar proofs Try methods 00%

Output: Query Sledgehammer Symbols

334,1 (11052/76028) (isabelle.isabelle.UTF-8-Isabelle) | nmro UG 160/512MB 237/990MB 10:20 PM

Appendix 3: Isabelle

The screenshot shows the Isabelle proof assistant interface. The main window displays a theorem script defining a class for cubical ω -categories with connections. The script includes several lemmas and a proof goal.

```

text <We define a class for cubical  $\omega$ -categories with connections.>

class cubical_omega_category_connections = cubical_omega_category + connection_ops +
  assumes conn_face1: "fFx j x  $\implies$   $\partial$  j  $\alpha$  ( $\Gamma$  j  $\alpha$  x) = x"
  and conn_face2: "fFx j x  $\implies$   $\partial$  (j + 1)  $\alpha$  ( $\Gamma$  j  $\alpha$  x) =  $\sigma$  j x"
  and conn_face3: "i  $\neq$  j  $\implies$  i  $\neq$  j + 1  $\implies$  fFx j x  $\implies$   $\partial$  i  $\alpha$  ( $\Gamma$  j  $\beta$  x) =  $\Gamma$  j  $\beta$  ( $\partial$  i  $\alpha$  x)"
  and conn_corner1: "fFx i x  $\implies$  fFx i y  $\implies$  DD (i + 1) x y  $\implies$   $\Gamma$  i tt (x  $\otimes_{\omega}$  (i + 1)  $\epsilon$  y) = ( $\Gamma$  i tt
  and conn_corner2: "fFx i x  $\implies$  fFx i y  $\implies$  DD (i + 1) x y  $\implies$   $\Gamma$  i ff (x  $\otimes_{\omega}$  (i + 1)  $\epsilon$  y) = ( $\Gamma$  i ff
  and conn_corner3: "j  $\neq$  i  $\wedge$  j  $\neq$  i + 1  $\implies$  fFx i x  $\implies$  fFx i y  $\implies$  DD j x y  $\implies$   $\Gamma$  i  $\alpha$  (x  $\otimes_{\omega}$  j y)
  and conn_fix: "fFx i x  $\implies$  fFx (i + 1) x  $\implies$   $\Gamma$  i  $\alpha$  x = x"
  and conn_zigzag1: "fFx i x  $\implies$   $\Gamma$  i tt x  $\otimes_{\omega}$  (i + 1)  $\epsilon$   $\Gamma$  i ff x = x"
  and conn_zigzag2: "fFx i x  $\implies$   $\Gamma$  i tt x  $\otimes_{\omega}$  i  $\epsilon$   $\Gamma$  i ff x =  $\sigma$  i x"
  and conn_conn_braid: "diffSup i j 2  $\implies$  fFx j x  $\implies$  fFx i x  $\implies$   $\Gamma$  i  $\alpha$  ( $\Gamma$  j  $\beta$  x) =  $\Gamma$  j  $\beta$  ( $\Gamma$  i  $\alpha$ 
  and conn_shift: "fFx i x  $\implies$  fFx (i + 1) x  $\implies$   $\sigma$  (i + 1) ( $\sigma$  i ( $\Gamma$  (i + 1)  $\alpha$  x)) =  $\Gamma$  i  $\alpha$  ( $\sigma$  (i + 1)

begin

lemma conn_face4: "fFx j x  $\implies$   $\partial$  j  $\alpha$  ( $\Gamma$  j ( $\neg$  $\alpha$ ) x) =  $\partial$  (j + 1)  $\alpha$  x"
  by (smt (z3) local.conn_face1 local.conn_zigzag2 local.face_comm_var local.locality local.pcomp
  
```

The interface includes a menu bar (File, Edit, Search, Markers, Folding, View, Utilities, Macros, Plugins, Help), a toolbar, and a status bar at the bottom showing the prover (cvc4) and various options like 'Isar proofs' and 'Try methods'.

Appendix 3: Isabelle

File Edit Search Markers Folding View Utilities Macros Plugins Help

CubicalCategories.thy (-Dossiers Tanguy/Cours/Thèse/Projets/Projets GitLab/isabelle/)

```

text <Next we define the class of cubical  $(\omega, 0)$ -categories with connections.>

class cubical_omega_zero_category_connections = cubical_omega_category_connections +
  assumes ri_inv: " $k \geq 1 \implies i \leq k - 1 \implies \text{dim\_bound } k \ x \implies \text{ri\_inv\_shell } k \ i \ x \implies \exists y. \text{ri\_inv } i \ x \ y$ "

begin

text <Finally, to show our axiomatisation at work we prove Proposition 2.4.7 from our companion pa
cell in an  $(\omega, 0)$ -category is ri-invertible for each natural number i. This requires some ba

lemma ri_inv_fix:
  assumes " $\forall x. \text{ri\_inv } i \ x$ "
  shows " $\exists y. \text{ri\_inv } i \ x \ y$ "
  by (metis assms icat.st_local local.face_compat_var local.icat.sscatml.l0_absorb)

lemma ri_inv2:
  assumes " $k \geq 1$ "
  assumes " $\text{dim\_bound } k \ x$ "
  and " $\text{ri\_inv\_shell } k \ i \ x$ "
  shows " $\exists y. \text{ri\_inv } i \ x \ y$ "
  
```

Provers: cvc4 verit z3 e spass vampire zipperposition | Isar proofs Try methods | Apply Cancel Locate 00%

Output: Query Sledgehammer Symbols

1598,1 (67709/76028) (isabelle.isabelle.UTF-8-Isabelle) | in mro UG | 05:44:31 PM | 496/1685 MiB | 10:21 PM

Appendix 3: Isabelle

The screenshot shows the Isabelle proof assistant interface. The main window displays a formal proof script for a lemma. The script is as follows:

```

lemma every dim k ri inv:
assumes "dim_bound k x"
shows "∃i. ∃y. ri_inv i x y" using <dim_bound k x>
proof (induct k arbitrary: x)
case 0
  thus ?case
  using ri_inv_fix by simp
next
case (Suc k)
  {fix i
   have "∃y. ri_inv i x y"
   proof (cases "Suc k ≤ i")
     case True
       thus ?thesis
       using Suc.prem1 ri_inv_fix by simp
     next
     case False
       {fix j α
        assume h: "j ≤ k ∧ j ≠ i"
        hence a: "dim_bound k (∑ j (k - j) (∂ j α x))"
          by (smt (z3) Suc.prem1 antisym_conv2 le_add_diff_inverse local.face_comm_var local.face_compat_var local.symcomp_face2 lo)
        have "∃y. ri_inv i (∂ j α x) y"
        proof (cases "j < i")
          case True
            obtain y where b: "ri_inv (i - 1) (∑ j (k - j) (∂ j α x)) y"
            using Suc.hyps a by force
            have c: "dim_bound k y"
            apply (rule_tac x = "∑ j (k - j) (∂ j α x)" in dim_ri_inv)
            using a b by simp_all
            hence d: "DD i (∂ j α x) (∂ j (k - j) y)"
            by (smt (verit) False True a b h icid.ts_compat le_add_diff_inverse local.iDst local.icid.stopp.ts_compat local.inv_sym)
            hence e: "DD i (∂ j (k - j) y) (∂ j α x)"
            by (smt (verit) False True b c dual_order.refl h icid.ts_compat le_add_diff_inverse local.iDst local.icid.stopp.ts_compat)
            have f: "∂ j α x ⊗1 ∂ j (k - j) y = ∂ j (k - j) (∑ j (k - j) (∂ j α x) ⊗1 (i - 1) y)"
            apply (subst inv_symcomp_comp4)
          }
        }
      }
  }

```

The interface includes a menu bar (File, Edit, Search, Markers, Folding, View, Utilities, Macros, Plugins, Help), a toolbar, and a status bar at the bottom showing the file path, line number (1598,1), and memory usage (121/512MB).