Graph transformation: a tool for designing geometric modeling operations

Romain Pascual
joint work with Pascale Le Gall, Hakim Belhaouari, and Agnès Arnould
April 11, 2023
CGAL's sew operation

```cpp
template<signed int n> void sew(DartDescriptor adart1, DartDescriptor adart2)
{
    CGAL_assertion(n == dimension);
    CGAL_assertion((n == 2) || (n == 3));
    size_type mark = get_nse_mark(n);
    CGAL::Map_dart_iterator_basic_of_involution<Graph, int>
        II(adart1, mark);
    CGAL::Map_dart_iterator_basic_of_involution<Graph, int>
        II(adart2, mark);
    for (; II.cont(); ++II, ++II)
    {
        Helper::template ProcessMarkedAttributes<
            CGAL::Internal::CGAL::MapAttributeFunction<Graph>,>
            run(II, mark);
    }
    negate_mark(mark);
    for (; II.cont(); II. rewind(); II.cont(); ++II, ++II)
    {
        basic_link_alpha<II, int, 3>;
    }
    negate_mark(mark);
    CGAL_assertion(is_whole_map_unmarked(mark));
    free_mark(mark);
}
Standard Utilisation
Domain-Specific Language

Standard Utilisation
Domain-Specific Language

Standard Utilisation

Automatic Inference?

Code Generation

Domain-Specific Language
Domain-Specific Language

Standard Utilisation

Automatic Inference?

Code Generation

Domain-Specific Language

n0

n1

<0, 1>

<0, 1>

<0, 1>

n0

n1

<0, 1>
Jerboa’s DSL

▶ Main characteristics:
  • Gmaps\(^1\)
  • Syntax analyzer\(^2\)

▶ Successful applications:
  • Plant growth
  • Architecture
  • Spring-mass
  • Geology

\(^1\)Poudret et al. 2008
\(^2\)Belhaouari et al. 2014
A sneak peek at Jerboa’s language
A sneak peek at Jerboa’s language
A sneak peek at Jerboa’s language
Running example: face triangulation
Standard Approach

Automatic Inference?

Code Generation

R. Pascual
1. Gmaps
1. Gmaps

2. Modeling operations
1. Gmaps
2. Modeling operations
3. Inference
Generalized maps

Geometric objects are represented with embedded generalized maps.
The category of graphs

A graph $G = (V, E, s, t)$:

- a set of nodes $V$,
- a set of arcs $E$,
- a source function $s : E \rightarrow V$,
- a target arrow $t : E \rightarrow V$,
The category of graphs

A graph $G = (V, E, s, t)$:
- a set of nodes $V$,
- a set of arcs $E$,
- a source function $s : E \to V$,
- a target arrow $t : E \to V$,

$\rightarrow$ Decorated with labels, types, and attributes.

A morphism $G \to H$:
- a node function $V_G \to V_H$,
- an arc function $E_G \to E_H$, preserving structure.
Generalized maps\textsuperscript{1}

\textsuperscript{1}Damiand et al. 2014.
Generalized maps\textsuperscript{1}

Color legend: 0, 1, 2.

\textsuperscript{1}Damiand et al. 2014.
Generalized maps\textsuperscript{1}

Color legend: 0, 1, 2.

- Topology: graph structure

\textsuperscript{1}Damiand et al. 2014.
Generalized maps\textsuperscript{1}

- Topology: graph structure
- Geometry: node attributes

Color legend: 0, 1, 2.

\textsuperscript{1}Damiand et al. 2014.
Orbits and topological cells

Orbit (encode topological cell):
Graph induced by a subset $\langle o \rangle \subseteq [0, n]$ of dimensions.

- positions on vertices (orbits $\langle 1, 2 \rangle$).

Color legend: 0, 1, 2.
Orbits and topological cells

Orbit (encode topological cell):
Graph induced by a subset \( \langle o \rangle \subseteq [0, n] \) of dimensions.

- positions on vertices (orbits \( \langle 1, 2 \rangle \)).
- colors on faces (orbits \( \langle 0, 1 \rangle \)).

Color legend: 0, 1, 2.
Formalizing modeling operations

- Operations on Gmaps are designed as graph rewriting rules.
Graph transformation rules

\[ L \xrightarrow{\text{R}} K \xrightarrow{\text{R}} R \]

Graph transformation rules

1Rozenberg 1997; Ehrig et al. 2006; Heckel et al. 2020.
Rewriting Gmaps
Orbit rewriting
Orbit rewriting

\[\begin{array}{ccc}
<0, 1>_{n0} & \rightarrow & <0, _>_{n0} \\
<_, 2>_{n1} & \rightarrow & <1, 2>_{n2}
\end{array}\]

\[\begin{array}{ccc}
\text{a} & \text{b} & \text{c} \\
\text{d} & \text{e} & \text{f} \\
\text{g} & \text{h}
\end{array}\]

\[\begin{array}{ccc}
\text{a} & \text{b} & \text{c} \\
\text{d} & \text{e} & \text{f} \\
\text{g} & \text{h}
\end{array}\]

\[r\]
Orbit rewriting

\[ \langle 0, 1 \rangle \quad \rightarrow \quad \langle 0, \_ \rangle \quad \rightarrow \quad \langle \_, 2 \rangle \quad \rightarrow \quad \langle 1, 2 \rangle \]

- \( n_0 \)
- \( n_1 \)
- \( n_2 \)

\[
\begin{align*}
\text{a} & \quad \text{b} \\
\text{c} & \quad \text{d} \\
\text{e} & \quad \text{fg} \\
\text{h} &
\end{align*}
\]

\[
\begin{align*}
\text{a} & \quad \text{b} \\
\text{c} & \quad \text{d} \\
\text{e} & \quad \text{fg} \\
\text{h} &
\end{align*}
\]
Orbit rewriting

\[ \langle 0, 1 \rangle \rightarrow \langle 0, \_ \rangle \rightarrow \langle \_, 2 \rangle \rightarrow \langle 1, 2 \rangle \]

\[ n_0 \rightarrow n_0 \rightarrow n_1 \rightarrow n_2 \]

\[ \text{R. Pascual} \]
Orbit rewriting

\[
\begin{align*}
&<0, 1>  \\
n0
\end{align*}
\rightsquigarrow

\begin{align*}
&<0, _>  \\
n0
\end{align*}

\begin{align*}
&<_, 2>  \\
n1
\end{align*}

\begin{align*}
&<1, 2>  \\
n2
\end{align*}

\begin{figure}
\centering
\begin{tikzpicture}
\node (a) at (0,0) {a};
\node (b) at (1,0) {b};
\node (h) at (0,1) {h};
\draw[->] (a) -- (b);
\draw[->] (h) -- (b);
\end{tikzpicture}
\end{figure}

\begin{figure}
\centering
\begin{tikzpicture}
\node (a) at (0,0) {a};
\node (b) at (1,0) {b};
\node (h) at (0,1) {h};
\draw[->] (a) -- (b);
\draw[->] (h) -- (b);
\end{tikzpicture}
\end{figure}

\begin{figure}
\centering
\begin{tikzpicture}
\node (a) at (0,0) {a};
\node (b) at (1,0) {b};
\node (c) at (0,1) {c};
\node (d) at (1,1) {d};
\node (e) at (0,2) {e};
\node (f) at (1,2) {f};
\node (g) at (0,3) {g};
\node (h) at (0,4) {h};
\draw[->] (a) -- (b);
\draw[->] (c) -- (d);
\draw[->] (e) -- (f);
\draw[->] (g) -- (h);
\end{tikzpicture}
\end{figure}

\begin{figure}
\centering
\begin{tikzpicture}
\node (a) at (0,0) {a};
\node (b) at (1,0) {b};
\node (c) at (0,1) {c};
\node (d) at (1,1) {d};
\node (e) at (0,2) {e};
\node (f) at (1,2) {f};
\node (g) at (0,3) {g};
\node (h) at (0,4) {h};
\draw[->] (a) -- (b);
\draw[->] (c) -- (d);
\draw[->] (e) -- (f);
\draw[->] (g) -- (h);
\end{tikzpicture}
\end{figure}
Orbit rewriting

\[ \langle 0, 1 \rangle \rightarrow \langle 0, \_ \rangle \rightarrow \langle \_, 2 \rangle \rightarrow \langle 1, 2 \rangle \]

\[ n_0 \rightarrow n_0 \rightarrow n_1 \rightarrow n_2 \]

\[ a \rightarrow b \rightarrow h \rightarrow n_0 \rightarrow n_2 \]

\[ a \rightarrow b \rightarrow h \rightarrow \langle 0, 1 \rangle \rightarrow \langle 1, 2 \rangle \]

\[ r \rightarrow \]

\[ a \rightarrow b \rightarrow c \rightarrow d \rightarrow e \rightarrow f \rightarrow g \rightarrow h \]

R. Pascual

APR Seminar

April 11, 2023 15 / 45
Orbit rewriting
Orbit rewriting

R. Pascual
Orbit rewriting

\[
\begin{array}{c}
<0, 1> \\
n_0
\end{array}
\quad \rightarrow 
\begin{array}{c}
<0, _> \\
n_0
\end{array}
\quad 1 \\
\begin{array}{c}
<_, 2> \\
n_1
\end{array}
\quad 0 \\
\begin{array}{c}
<1, 2> \\
n_2
\end{array}
\end{array}
\]
The complete construction
The complete construction

Input 1: Rule scheme
Output: Gmap

Input 2: Gmap

Output: Gmap

$1_{\mathcal{W} \times \mathcal{D}}$

$!_{E_{\mathcal{D}}(P_v)}$

$E_{\mathcal{D}}(P_v)$

$E_{\mathcal{D}}$

$p_v$

$\mathcal{P}(\mathcal{G})$

$L \leftarrow K \rightarrow R$

$L \leftarrow E_{\mathcal{D}}(P_v) \times L \leftarrow E_{\mathcal{D}}(P_v) \times K \rightarrow E_{\mathcal{D}}(P_v) \times R$

$\pi_{\mathcal{D}}$

$\iota(L, P_v)$

$\iota(K, P_v)$

$\iota(R, P_v)$

$m$

$\mathcal{G}$

Input 2: Gmap

$\mathcal{D}$

$\mathcal{H}$

Output: Gmap
The complete construction

'Standard' DPO step
Modifying geometric values

1Bellet et al. 2017.
Modifying geometric values

Algebraic data types:
- point3D, colorRGB, ...
- add, middle, scale, ...

Modifying geometric values

Extended with topological operators:

- Neighbor operator:
  - $a@0@1@0.position = f.position = C$
  - $a@1@0.color = c.color = \bullet$

---

Modifying geometric values

Extended with topological operators:

- Neighbor operator:
- Collect operator:
  
  ▶ $\text{position}_{\langle 0,1 \rangle}(a) = \{A, B, C, D\}$
  
  ▶ $\text{color}_{\langle 0,1 \rangle}(a) = \{\text{●}\}$

$^1$Bellet et al. 2017.
Extension to schemes
Extension to schemes

\[
\begin{align*}
<0, 1> &\quad \xrightarrow{\text{black arrow}} <0, \_> \\
&\quad \xrightarrow{1} <\_, 2> \\
&\quad \xrightarrow{0} <1, 2> \text{ position}
\end{align*}
\]
Extension to schemes

Diagram illustrating the extension of schemes with arrows and nodes labeled.

- \(<0, 1>\) to \(<0, \_>\)
- \(<\_, 2>\) to \(<1, 2>\)
- Nodes and edges representing the extension process.
Extension to schemes

\[ \frac{1}{4}(A + B + C + D) \]
Extension to schemes

\[
\frac{1}{4}(A + B + C + D) = \text{middle}(\{A, B, C, D\})
\]
Extension to schemes

\[
\frac{1}{4}(A + B + C + D) = \text{middle}(\{A, B, C, D\}) = \text{middle}(\text{position}_{\langle 0, 1 \rangle}(a))
\]
Extension to schemes

\[
\frac{1}{4}(A + B + C + D) = \text{middle}(\{A, B, C, D\}) \\
= \text{middle}(\text{position}_{0,1}(a)) \\
= \text{middle}(\text{position}_{0,1}(n0))
\]
Consistency preservation

Modifying a well-formed object should produce a well-formed object.

Feedback to the rule designer.
Consistency preservation

Modifying a well-formed object should produce a well-formed object.

Feedback to the rule designer.

▶ Topological inconsistencies

▶ Geometric inconsistencies
Breaking the topological consistency

Constraint: 0202 paths should be cycles.
Breaking the topological consistency

**Constraint:** 0202 paths should be cycles.

![Diagram showing topological consistency](image)
Constraint: 0202 paths should be cycles.
Breaking the geometric consistency

Constraint: nodes in a $\langle 0, 1 \rangle$-orbit should have the same color.

mix(a.color, b.color)
Breaking the geometric consistency

Constraint: nodes in a $\langle 0, 1 \rangle$-orbit should have the same color.

$$\text{mix}(a.\text{color}, b.\text{color})$$
Breaking the geometric consistency

Constraint: nodes in a \langle 0, 1 \rangle-orbit should have the same color.

Rule completion

\text{mix}(a.\text{color}, b.\text{color})
Formalizing Jerboa’s DSL

- Jerboa’s DSL with categorical constructions: products, attributes, completion.
- Weaker consistency conditions.
- Unified framework to study generalized and oriented maps.


Agnès Arnould et al. (2022). “Preserving consistency in geometric modeling with graph transformations”. In: Mathematical Structures in Computer Science. DOI: 10.1017/S0960129522000226
Formalizing Jerboa’s DSL

- Jerboa’s DSL with categorical constructions: products, attributes, completion.
- Weaker consistency conditions.
- Unified framework to study generalized and oriented maps.

Main lesson:
a DSL allows for the safe design of geometric modeling operations.

- Agnès Arnould et al. (2022). “Preserving consistency in geometric modeling with graph transformations”. In: Mathematical Structures in Computer Science. DOI: 10.1017/S0960129522000226
Inferring geometric modeling operations

- Retrieving the operation described by an example.
Instances
Folding a joint representation of the rule

Topological folding algorithm: graph traversal folding nodes and arcs.

Input:
- two partial Gmaps
- preservation links
- a dart
- an orbit type.

 Orbit type $\langle 0, 1 \rangle$ and dart $a_0$. 

R. Pascual
APR Seminar
April 11, 2023
Folding a joint representation of the rule

Topological folding algorithm: graph traversal folding nodes and arcs.

Input:
- two partial Gmaps
- preservation links
- a dart
- an orbit type.

Orbit type $\langle 0, 1 \rangle$ and dart $a_0.$
Folding a joint representation of the rule

Color legend: 0, 1, 2, $\kappa$. 

Topological folding algorithm: graph traversal folding nodes and arcs.

Input:
- two partial Gmaps
- preservation links
- a dart
- an orbit type.

$\langle 0, 1 \rangle$ and dart $a_0$. 

R. Pascual

APR Seminar

April 11, 2023
Folding a joint representation of the rule

Color legend: 0, 1, 2, $\kappa$.

Topological folding algorithm: graph traversal folding nodes and arcs.

Input:
- two partial Gmaps
- preservation links
- a dart
- an orbit type.

- Orbit type $\langle 0, 1 \rangle$ and dart $a0$. 
Execution

Hook (orbit $\langle 0, 1 \rangle$).

Color legend: 0, 1, 2, $\kappa$. 
Execution

Folding the arcs.

Color legend: 0, 1, 2, \( \kappa \).
Execution

Color legend: 0, 1, 2, κ.

Folding a node.

<0, 1> n0 \rightarrow K <0, _> n1
The algorithm terminates.

Color legend: 0, 1, 2, $\kappa$. 
Splitting the joint representation.

Color legend: 0, 1, 2, $\kappa$. 
Correctness: The algorithm returns a topological folding of the rule if it exists and halts otherwise.


What about cases where we cannot fold the rule? Orbit $\langle 0, 1, 2 \rangle$. 
Correctness: The algorithm returns a topological folding of the rule if it exists and halts otherwise.


What about cases where we cannot fold the rule? Orbit $\langle 0, 1, 2 \rangle$. 
Correctness: The algorithm returns a topological folding of the rule if it exists and halts otherwise.


What about cases where we cannot fold the rule? Orbit $\langle 0, 1, 2 \rangle$. 
Hypothesis: Affine combinations of positions.
Hypothesis: Affine combinations of positions.

For each vertex in $C$, we want a position $p$ expressed as:

$$p = \sum_{i=0}^{k} w_i p_i + t$$

where:

- $p$ : target position (known)
Hypothesis: Affine combinations of positions.

For each vertex in $C$, we want a position $p$ expressed as:

$$p = \sum_{i=0}^{k} w_i p_i + t$$

where:

- $p$ : target position (known)
- $p_i$: position of the initial vertex $i$ (known)
Hypothesis: Affine combinations of positions.

For each vertex in $C$, we want a position $p$ expressed as:

$$p = \sum_{i=0}^{k} w_i p_i + t$$

where:

- $p$: target position (known)
- $p_i$: position of the initial vertex $i$ (known)
- $w_i$: weight (unknown)
- $t$: translation (unknown)
Method (inference of positions)

Hypothesis: Affine combinations of positions.

For each vertex in $C$, we want a position $p$ expressed as:

$$p = \sum_{i=0}^{k} w_i p_i + t$$

where:

- $p$: target position (known)
- $p_i$: position of the initial vertex $i$ (known)
- $w_i$: weight (unknown)
- $t$: translation (unknown)
Need for abstraction on schemes

\[(w_i)_{0 \leq i \leq k} \text{ such that: } p = \sum_{i=0}^{k} w_i p_i + t\]
Need for abstraction on schemes

\[(w_i)_{0 \leq i \leq k} \text{ such that: } p = \sum_{i=0}^{k} w_i p_i + t\]

Rule schemes abstract topological cells.
Need for abstraction on schemes

$$\left(w_i\right)_{0 \leq i \leq k}$$ such that: $$p = \sum_{i=0}^{k} w_ip_i + t$$

Rule schemes abstract topological cells.

**Issue:** Darts share the same expression.

Solution: Exploit the topology.

▶ Use points of interest.
Need for abstraction on schemes

\[(w_i)_{0 \leq i \leq k} \text{ such that: } p = \sum_{i=0}^{k} w_i p_i + t\]

Rule schemes abstract topological cells.

**Issue:** Darts share the same expression.

**Solution:** Exploit the topology.

- Use points of interest.
Points of interest

- \( v \): vertex
- \( e \): edge midpoint
- \( f \): face barycenter
- \( s \): volume barycenter
- \( cc \): CC barycenter
Points of interest

- \( p_v \): vertex
- \( p_e \): edge midpoint
- \( p_f \): face barycenter
- \( p_s \): volume barycenter
- \( p_{cc} \): CC barycenter
Points of interest

with

- \( p_v \) : vertex

\[
p_v = \text{middle}(position_{1,2,3}(d))
\]
Points of interest

with

- $p_v$ : vertex
- $p_e$ : edge midpoint

\[ p_e = \text{middle}(\text{position}_{\{0,2,3\}}(d)) \]
Points of interest

with

- $p_v$: vertex
- $p_e$: edge midpoint
- $p_f$: face barycenter

$$p_f = \text{middle}(\text{position}_{0,1,3}(d))$$
Points of interest

with

- $p_v$: vertex
- $p_e$: edge midpoint
- $p_f$: face barycenter
- $p_s$: volume barycenter

\[ p_s = \text{middle}(\text{position}_{\langle 0,1,2 \rangle}(d)) \]
Points of interest

with

- $p_v$ : vertex
- $p_e$ : edge midpoint
- $p_f$ : face barycenter
- $p_s$ : volume barycenter
- $p_{cc}$ : CC barycenter

\[
p_{cc} = \text{middle}(\text{position}_{\langle 0,1,2,3 \rangle}(d))
\]
Points of interest

with

- $p_v$: vertex
- $p_e$: edge midpoint
- $p_f$: face barycenter
- $p_s$: volume barycenter
- $p_{cc}$: CC barycenter

The system becomes:

$$p = w_v p_v + w_e p_e + w_f p_f + w_s p_s + w_{cc} p_{cc} + t$$
Position of $n2$ as a function of $n0$. 

- One equation per dart (8 darts).
- Split per coordinate (on $x$, $y$, $z$).
- 24 equations and 8 variables.
Position of $n_2$ as a function of $n_0$.

$$n_2.position = w_v n_0.p_v + w_e n_0.p_e + w_f n_0.p_f + w_s n_0.p_s + w_{cc} n_0.p_{cc} + t$$
Position of $n2$ as a function of $n0$.

- One equation per dart (8 darts).

\[
n2.position = w_v n0.p_v + w_e n0.p_e + w_f n0.p_f + w_s n0.p_s + w_{cc} n0.p_{cc} + t
\]

- One equation per dart (8 darts).
Position of $n2$ as a function of $n0$.

- One equation per dart (8 darts).
- Split per coordinate (on $x$, $y$, $z$).

$$n2\.position = w_v n0\.p_v + w_e n0\.p_e + w_f n0\.p_f + w_s n0\.p_s + w_{cc} n0\.p_{cc} + t$$

- **vertex**
- **edge**
- **face**
- **volume**
- **cc**
Position of $n_2$ as a function of $n_0$.

- One equation per dart (8 darts).
- Split per coordinate (on $x$, $y$, $z$).
- 24 equations and 8 variables.

$$n_2\text{.position} = \underbrace{w_v n_0\text{.p}_v}_{\text{vertex}} + \underbrace{w_e n_0\text{.p}_e}_{\text{edge}} + \underbrace{w_f n_0\text{.p}_f}_{\text{face}} + \underbrace{w_s n_0\text{.p}_s}_{\text{volume}} + \underbrace{w_{cc} n_0\text{.p}_{cc}}_{\text{cc}} + t$$
Position of $n_2$ as a function of $n_0$.

- One equation per dart (8 darts).
- Split per coordinate (on $x$, $y$, $z$).
- 24 equations and 8 variables.

$$n_2.\text{position} = w_v n_0.p_v + w_e n_0.p_e + w_f n_0.p_f + w_s n_0.p_s + w_{cc} n_0.p_{cc} + t$$

- CSP, solvers used: OR-Tools (Google), Z3 (Microsoft)
Solving the barycentric triangulation

Global equation:

\[ n2.\text{position} = w_v n0.p_v + w_e n0.p_e + w_f n0.p_f + w_s n0.p_s + w_{cc} n0.p_{cc} + t \]
Solving the barycentric triangulation

Global equation:

\[ n2.\text{position} = w_v n0.\text{p}_v + w_e n0.\text{p}_e + w_f n0.\text{p}_f + w_s n0.\text{p}_s + w_{cc} n0.\text{p}_{cc} + t \]

Generated system

\[
\begin{align*}
(0.5; 0.5) &= w_v \ast (0; 0) + w_e \ast (0.5; 0) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v \ast (1; 0) + w_e \ast (0.5; 0) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v \ast (1; 0) + w_e \ast (1; 0.5) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v \ast (1; 1) + w_e \ast (1; 0.5) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty)
\end{align*}
\]
Solving the barycentric triangulation

Global equation:
\[ n2.position = w_v n0.p_v + w_e n0.p_e + w_f n0.p_f + w_s n0.p_s + w_{cc} n0.p_{cc} + t \]

Generated system
\[
\begin{align*}
(0.5; 0.5) &= w_v * (0; 0) + w_e * (0.5; 0) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v * (1; 0) + w_e * (0.5; 0) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v * (1; 0) + w_e * (1; 0.5) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v * (1; 1) + w_e * (1; 0.5) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty)
\end{align*}
\]

Solution found:
- \( w_v = 0.0 \)
- \( w_e = 0.0 \)
- \( w_f = 1.0 \)
- \( w_s = 0.0 \)
- \( w_{cc} = 0.0 \)
- \( t = (0.0, 0.0) \)
JerboaStudio and applications

- Implementation of the inference mechanism in Jerboa.
JerboaStudio
Example inspired from geology

Before

After

Operation

Inference time: $\sim 3$ ms
Example inspired from geology
Example inspired from geology
Example inspired from geology (part 2)

Before

After

Both positions and colors.
Inference time: ∼ 26 ms for the topology,
∼ 549 ms for the embedding expressions
Example inspired from geology
Example inspired from geology
Doo-Sabin subdivision\textsuperscript{1}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{doo-sabin-subdivision.png}
\caption{Doo-Sabin subdivision process.}
\end{figure}

\textsuperscript{1}Doo et al. 1978.
Doo-Sabin subdivision

1

Doo et al. 1978.

R. Pascual

APR Seminar

April 11, 2023

38 / 45
Doo-Sabin subdivision

$\langle 0, 1, 2 \rangle \rightarrow \langle 0, 1, \_ \rangle \rightarrow \langle 0, \_ \_ \rangle \rightarrow \langle \_, 0 \_ \rangle \rightarrow \langle \_, 1, 0 \rangle$

$\rightarrow 3^{rd}$ iteration:

$\rightarrow \langle 0, 1, 2 \rangle \rightarrow \langle 0, 1, \_ \rangle \rightarrow \langle 0, \_ \_ \rangle \rightarrow \langle \_, 0 \_ \rangle \rightarrow \langle \_, 1, 0 \rangle$

$\rightarrow \langle 0, 1, 2 \rangle \rightarrow \langle 0, 1, \_ \rangle \rightarrow \langle 0, \_ \_ \rangle \rightarrow \langle \_, 0 \_ \rangle \rightarrow \langle \_, 1, 0 \rangle$

---

$^1$Doo et al. 1978.
Menger \((2, 2, 2)\)^1

^1\text{Richaume et al. 2019.}
Menger \((2, 2, 2)\)\(^1\)

\(^1\)Richaume et al. 2019.
Menger \((2, 2, 2)\)^1

\(^1\text{Richaume et al. 2019.}\)
Edge cases

▶ Von Koch’s snowflake generated with L-systems

▶ Inferred:
Edge cases

- Von Koch’s snowflake generated with L-systems

- Inferred:
JerboaStudio’s architecture

Editor
- Object specifications
- Dimensions and embeddings
- Creation of rules
  Quad subdivision, face triangulation, ...
- Static analysis

Embedding Libraries
- 3D Coordinates, RGB Colors, ...

Jerboa Kernel
- Rule application engine
- Quad subdivision, face triangulation, ...

Generated Modeler Kernel

Generic Viewer
- Load
- Save
- Apply Operations

Automated
User input
Generic
Specific
JerboaStudio’s architecture

Editor
- Object specifications
  - Dimensions and embeddings
- Creation of rules
  - Quad subdivision, face triangulation, ...
- Static analysis

Embedding Libraries
- 3D Coordinates, RGB Colors, ...

Jerboa Kernel
- Rule application engine
- Quad subdivision, face triangulation, ...

Generated Modeler Kernel

Generic Viewer
- Load
- Save
- Apply Operations

Inference Module
Conclusion

▶ Ongoing works and main contributions.
Towards local nested conditions?

Rules should be checked statically.
Towards local nested conditions?

Rules should be checked statically.

Calculus for constraint preserving and constraint guaranteeing rules.\(^1\)

\(^1\)Pennemann 2009.
Towards local nested conditions?

Rules should be checked statically.

Calculus for constraint preserving and constraint guaranteeing rules.¹

- Rely on global computations.
- Does not scale very well (with the size of the graphs).

¹Pennemann 2009.
Towards local nested conditions?

Rules should be checked statically.

Calculus for constraint preserving and constraint guaranteeing rules.\(^1\)

- Rely on global computations.
- Does not scale very well (with the size of the graphs).

\(\uparrow\) Collaboration with Nicolas Behr and Pascale Le Gall.

\(^1\)Pennemann 2009.
Towards a multi-cell query-replace approach?

Query-replace modifications with a text editor but for combinatorial structures.¹

¹Damiand et al. 2022
Towards a multi-cell query-replace approach?

Query-replace modifications with a text editor but for combinatorial structures.¹

How to extend the approach to multicell patterns?

¹Damiand et al. 2022
Towards a multi-cell query-replace approach?

Query-replace modifications with a text editor but for combinatorial structures.¹

How to extend the approach to multicell patterns?

► Collaboration with Guillaume Damiand and Vincent Nivoliers supported by the GDR IGRV.

¹Damiand et al. 2022
Formalization of the DSL
Formalization of the DSL

Graph transformation as a backbone for inferring...
Formalization of the DSL

Graph transformation as a backbone for inferring

Inference Module

~ 9000 LoC

R. Pascual  APR Seminar  April 11, 2023  45 / 45
Arnould, Agnès et al. (2022). “Preserving consistency in geometric modeling with graph transformations”. In: Mathematical Structures in Computer Science. DOI: 10.1017/S0960129522000226.


Damiand, Guillaume et al. (June 18, 2022). “Query-replace operations for topologically controlled 3D mesh editing”. In: Computers & Graphics. ISSN: 0097-8493. DOI: 10.1016/j.cag.2022.06.008.


