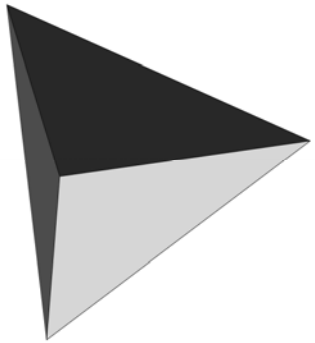
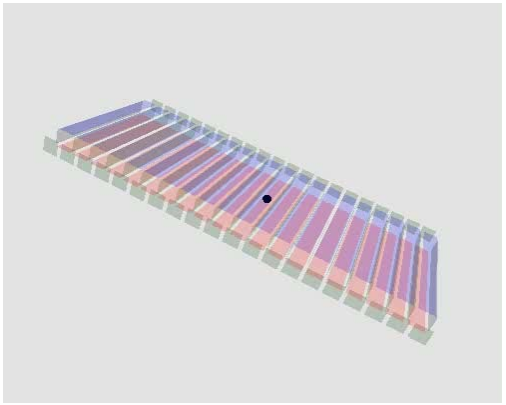
Topological rewriting, autonomous systems and the geometrization of programming



Antoine Spicher

MGS Project - Team **LIS**
IBISC – FRE 2873 CNRS
Université d'Évry



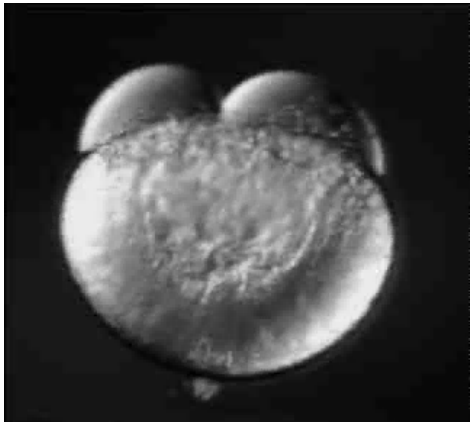
Problematic

- Unbounded computing media
  - ✓ In silicon
    FPGAs, claytronics, smart dust/matters, grid computing, Internet
  - ✓ Advances in chemistry/biology
    DNA/RNA computing, synthetic biology
  - → A potentially infinite amount of computing resources

- Spatially organized systems
  - ✓ Dealing with entities
    - spatially *structured*
      composed of elements organized by a neighborhood relationship
    - by an *intentional* way
      as a whole, no explicit scheme for elements iterations
  - ✓ How to program such a system?

Characteristics of spatial systems

- Dynamical systems with a dynamical structure
  - ✓ The evolution function cannot be defined *a priori*
  - ✓ State space has to be computed during the simulation
  - ✓ Example: embryogenesis



- Autonomic computing systems
  - ✓ Autonomous elements together with local decision mechanisms
  - ✓ Self-* behaviors

    Self-organization, self-healing, self-optimization, …

Outline

- MGS Project
  - ✓ Topological collections
  - ✓ Transformations
  - ✓ Examples of applications
- Spatial programming of a biosynthetic medium
  - ✓ Biosynthetic: a wetware
  - ✓ Algebraic programming
  - ✓ Discrete differential calculus
- Conclusion

Modeling and simulation of DS^2

Rewriting for modeling

Complex systems

Rewriting techniques

Modeling

State (space)

Hierarchical organizations

Dynamics/evolution

Local interaction

Trajectories

Definition

Data structure

Formal trees (terms)

Rewriting system

α => β, α: pattern, β: expression

Derivations

Simulation

Time managing

Discrete, event based: synchronous/asynchronous/…

Application

Rule application strategies

Maximal parallel/sequential/ stochastic/…

Modeling and simulation of DS^2

Rewriting for modeling

Complex systems

Modeling

State (space)

Hierarchical organizations

Arbitrary organizations

Dynamics/evolution

Local interaction

Trajectories

Simulation

Time managing

Discrete, event based: synchronous/asynchronous/…

Rewriting techniques

Definition

Data structure

Formal trees (terms)

What kind of data structures?

Rewriting system

α => β, α: pattern, β: expression

Derivations

Application

Rule application strategies

Maximal parallel/sequential/ stochastic/…

Modeling and simulation of DS²

Rewriting for modeling

Complex systems

Modeling

State (space)

Hierarchical organizations

Arbitrary organizations

Dynamics/evolution

Local interaction

Trajectories

Simulation

Time managing

Discrete, event based: synchronous/asynchronous/…

Rewriting techniques

Definition

Data structure

Formal trees (terms)

Topological collections

Transformation

α => β, α: pattern, β: expression

Derivations

Application

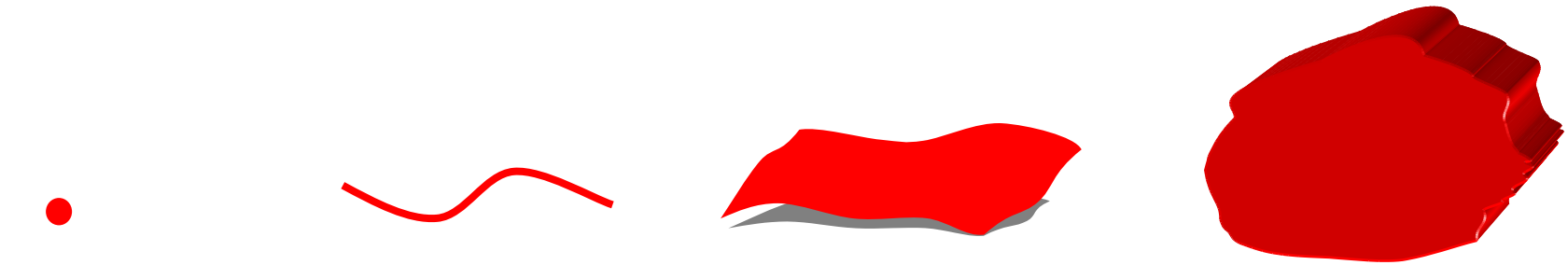
Rule application strategies

Maximal parallel/sequential/ stochastic/…

Topological collection

A topological collection is

- ✓ *Structure:* abstract cellular complex
  - A collection of topological cells

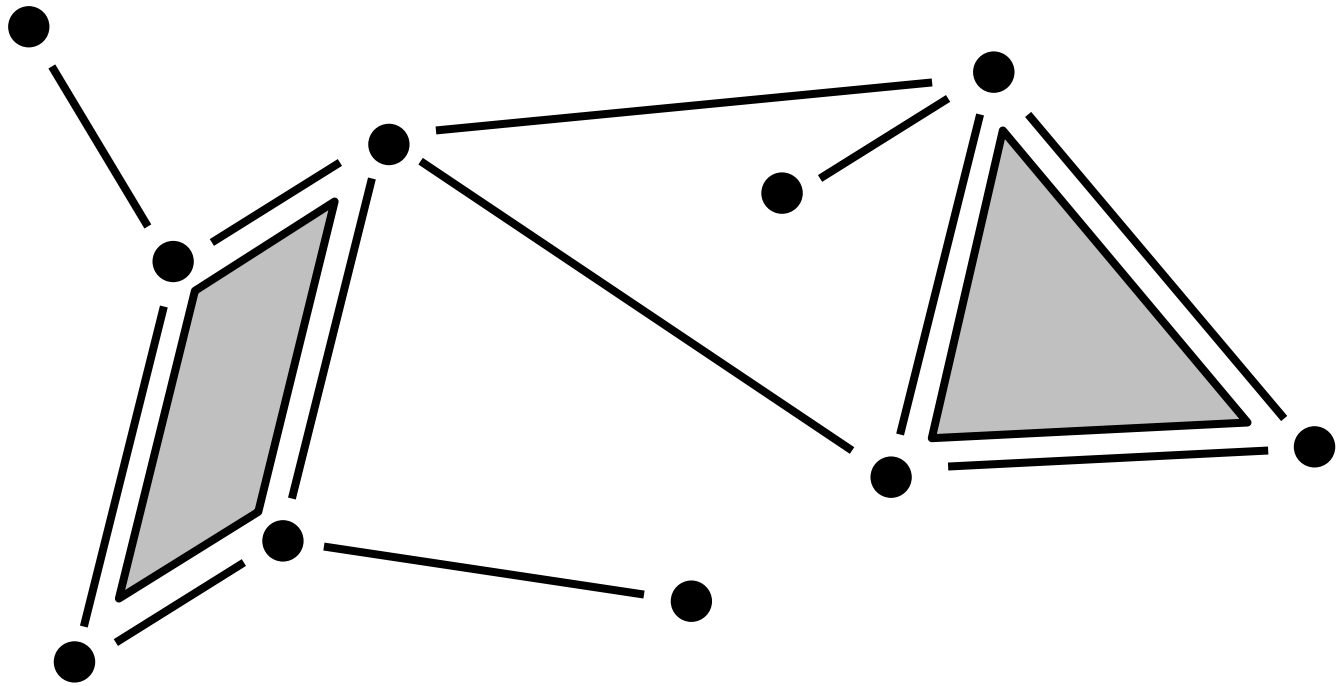


0-cell 1-cell 2-cell 3-cell

Topological collection

A topological collection is

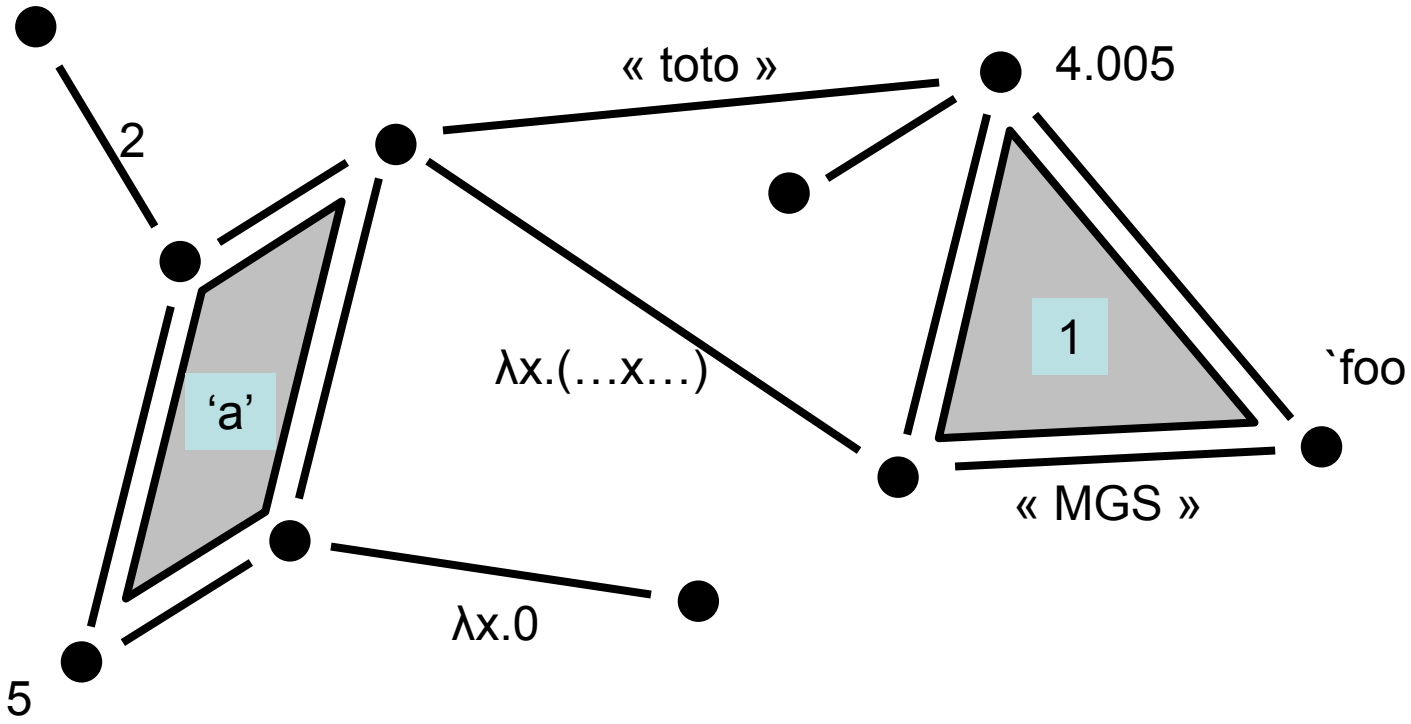
- ✓ *Structure:* abstract cellular complex
  - A collection of topological cells
  - The incidence relationship



Topological collection

A topological collection is

- ✓ *Structure:* abstract cellular complex
  - A collection of topological cells
  - The incidence relationship
- ✓ of *data:* associate a value with each cell



Transformation

- Handling topological collection
- Transformation
  - ✓ Function defined by case
  - ✓ Each case matches a sub-collection
  - ✓ A case = a pattern
- Local collection rewriting
  - ✓ **Matching** of a sub-collection **s**
  - ✓ **Computing** a new sub-collection **s'** as a function of **s**
  - ✓ **Substituting s** by **s'**

$$T = \begin{cases} rule_1 : pattern_1 \Rightarrow \text{expression}_1 \\ rule_2 : pattern_2 \Rightarrow \text{expression}_2 \\ \ldots \\ rule_n : pattern_n \Rightarrow \text{expression}_n \end{cases}$$

Transformation

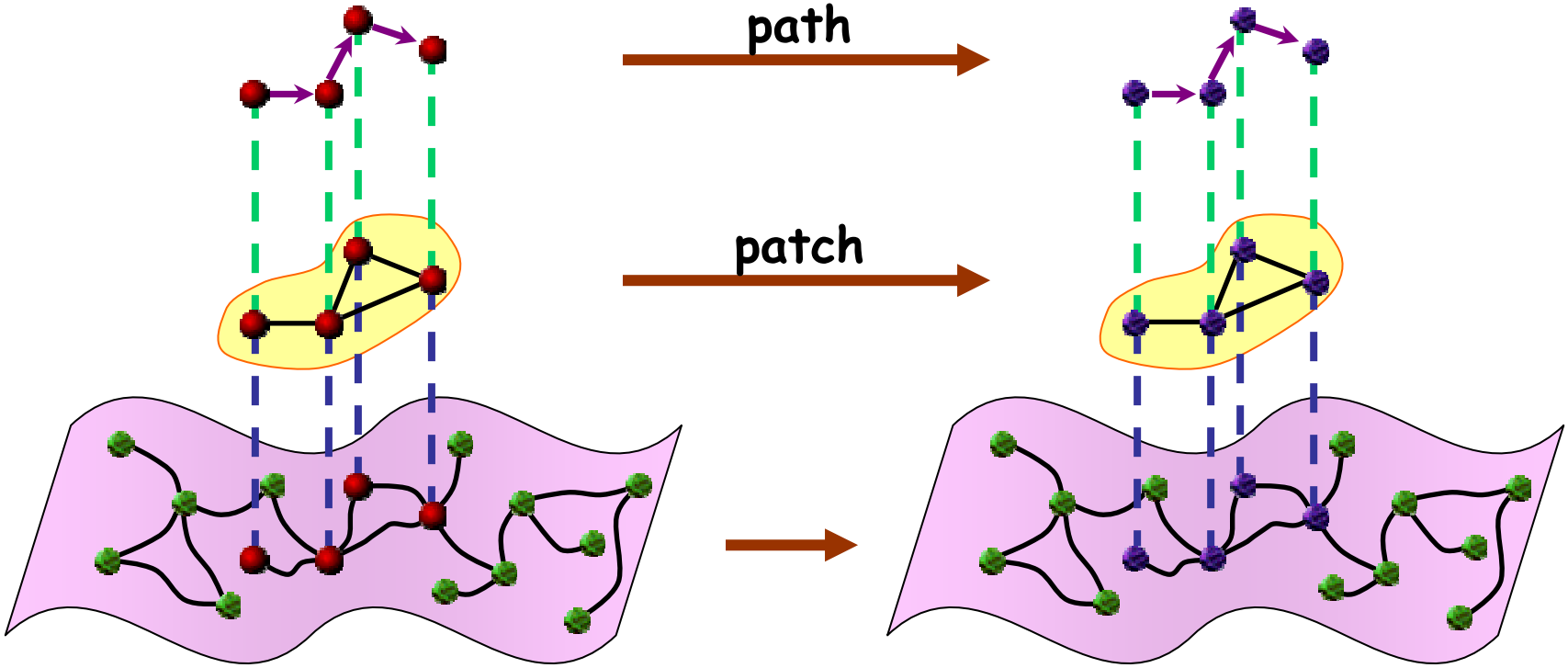
Transformations: sub-collection rewriting

- ✓ Path transformation *or* <n,p>-transformation (<n,p>-path)

  Concise but limited expressivity

- ✓ Patch transformation (arbitrary shape using (co)face relations)

  Longer but higher expressivity

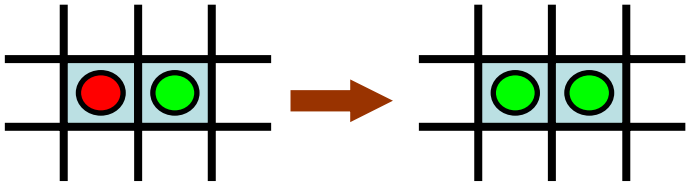


Transformation

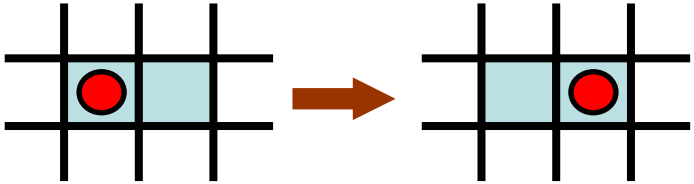
Two pattern languages

- ✓ Path pattern ➔ <n,p>-transformations

Example: DLA



aggregation

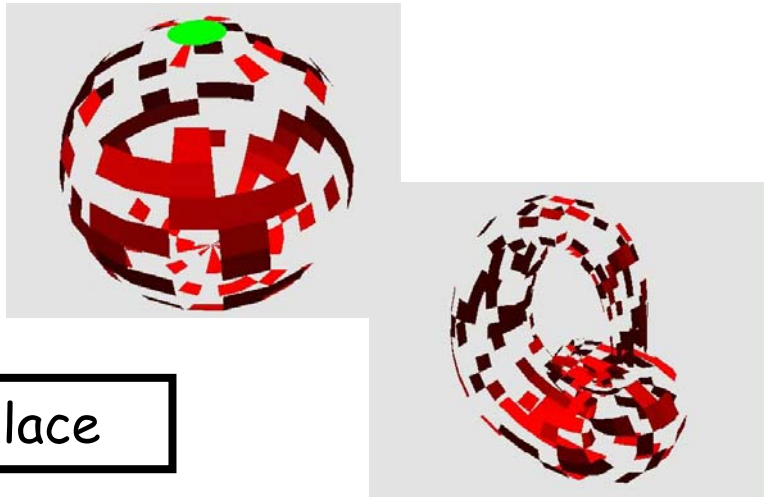


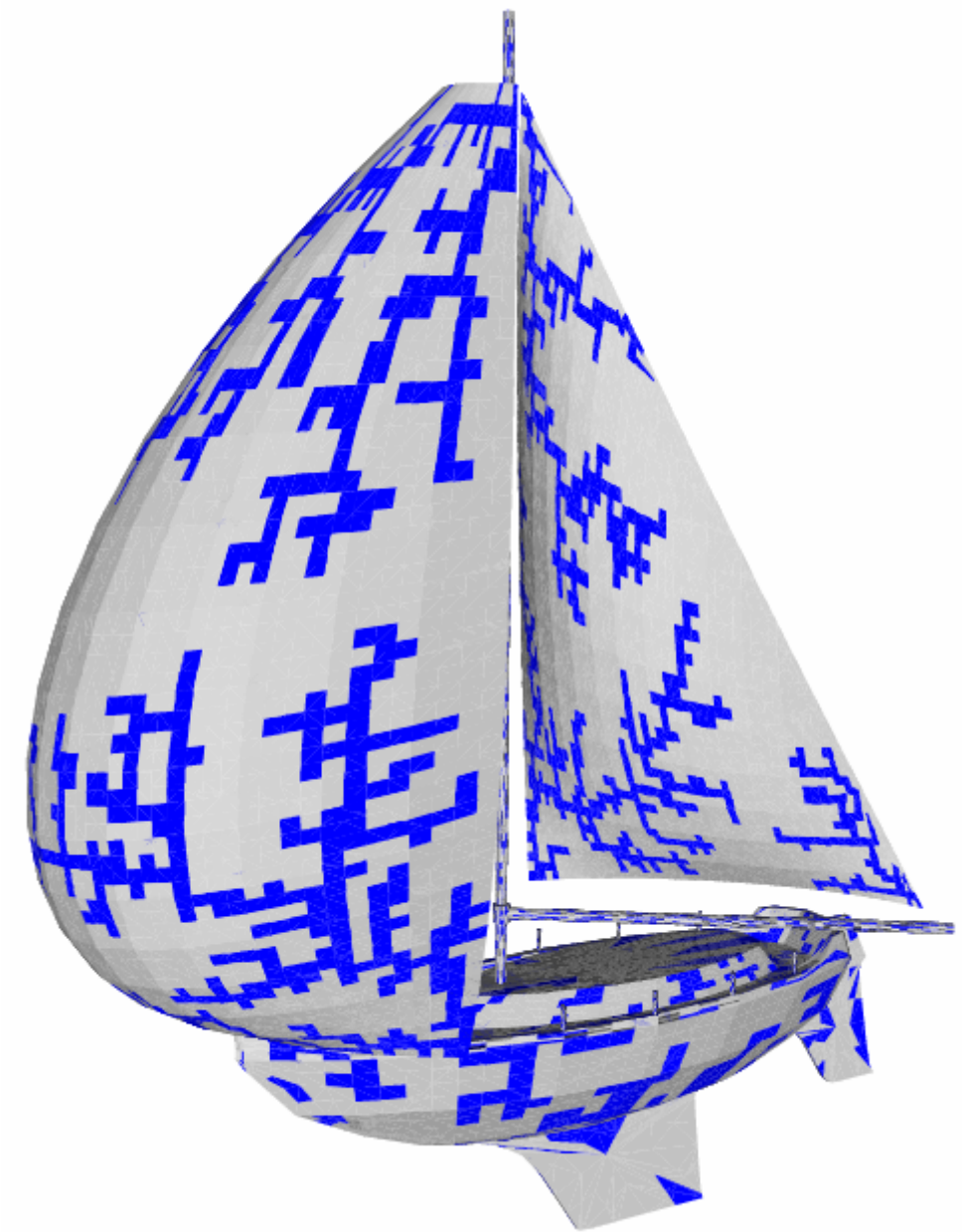
diffusion

```
trans <2,1> dla = {
   `red, `green   => `green, `green ;
   `red, <undef> => <undef>, `red
} ;;
```

<2,1>-neighborhood

empty place





Transformation

Two pattern languages

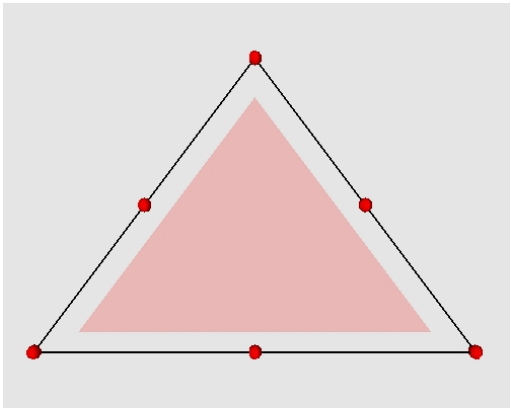
- ✓ Patch pattern ➔ patch transformations

Example: vertex insertion

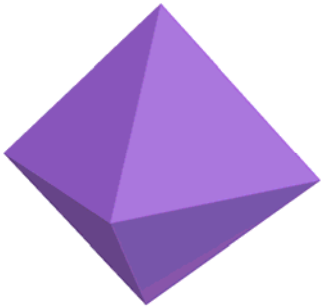

```
patch insert_vertex = {
 ~v1 < e:[dim=1] > ~v2
  =>
   `v :[dim=0, val = …                    ]
  `e1:[dim=1, faces=[^v1,`v], val= …]
   e2:[dim=1, faces=[^v2, `v], val= …]
} ;;
```

face of creation of an ege between v1 and `v

not consumed

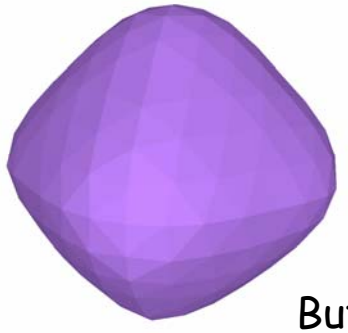


Mesh refinement

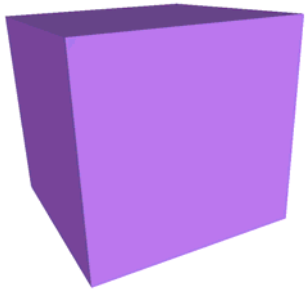


Triangular mesh

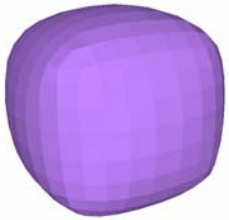
Loop



Butterfly

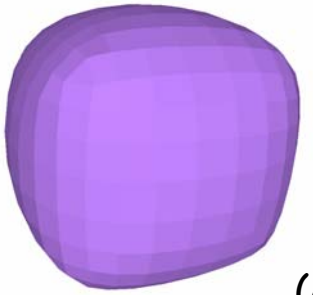


Quadrangular mesh



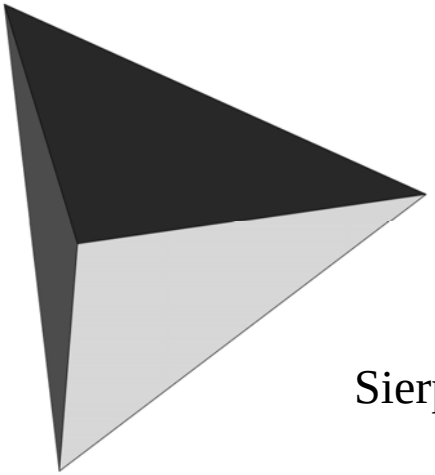
Catmull-Clark

Kobbelt

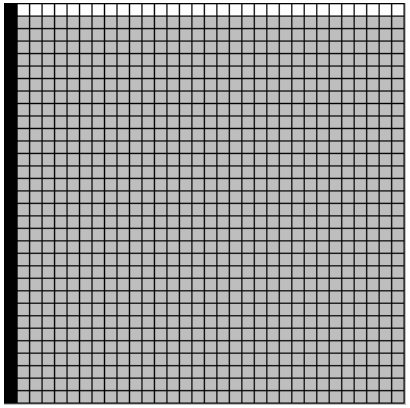


Doo-Sabin
(chanfreinage)

Self-(dis)assembly of fractals

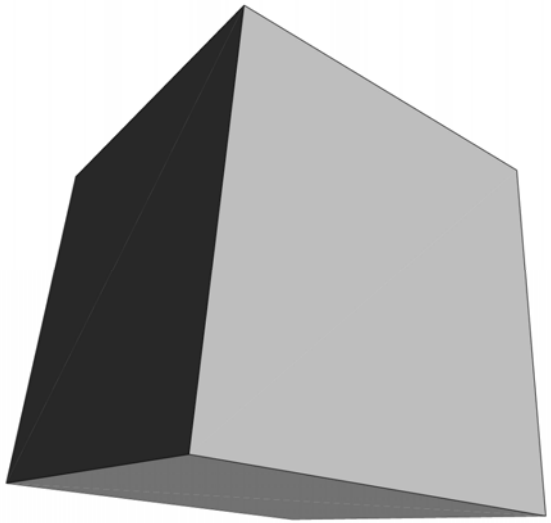


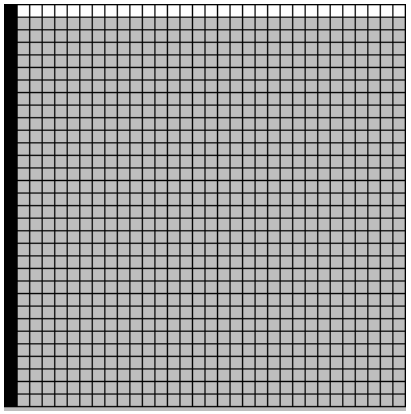
Sierpinski sponge



Maximal parallel strategy

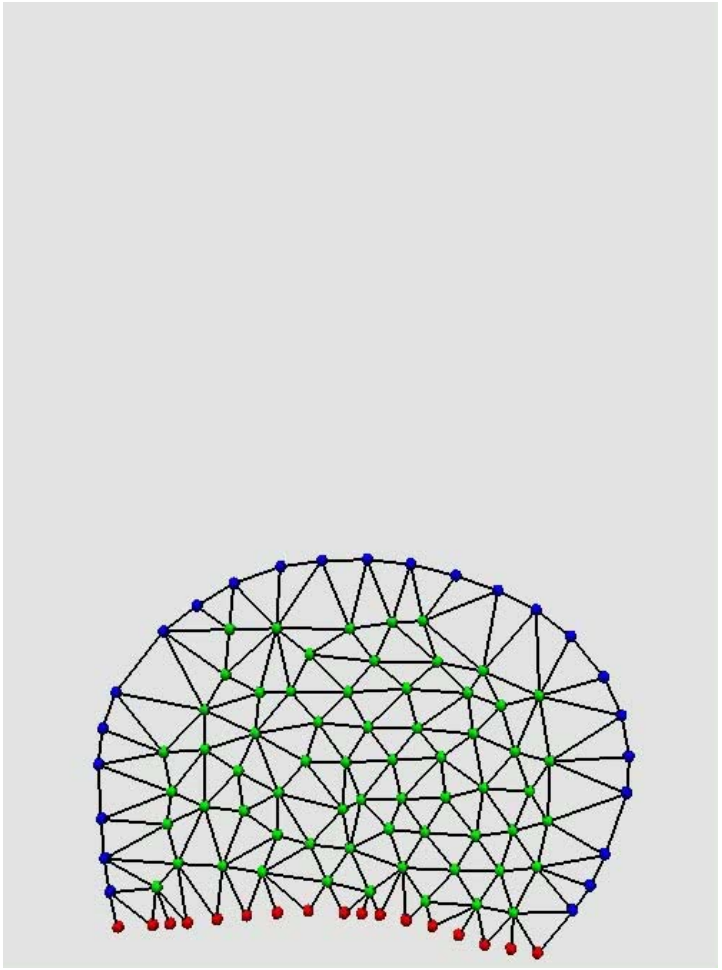
Menger sponge





Stochastic strategy

Dynamical structures in biology

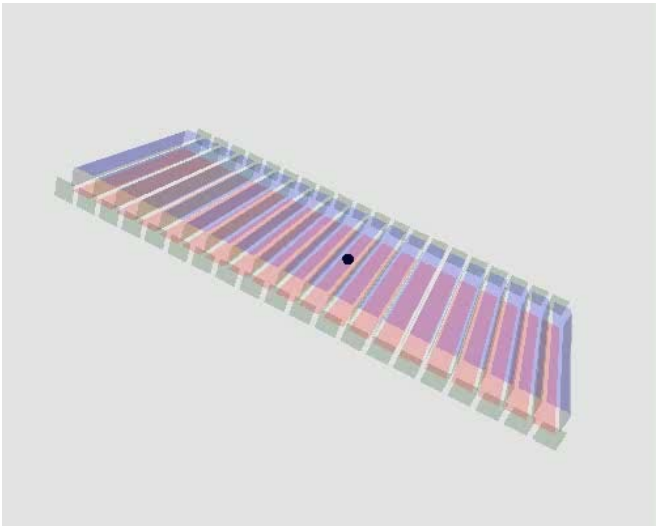


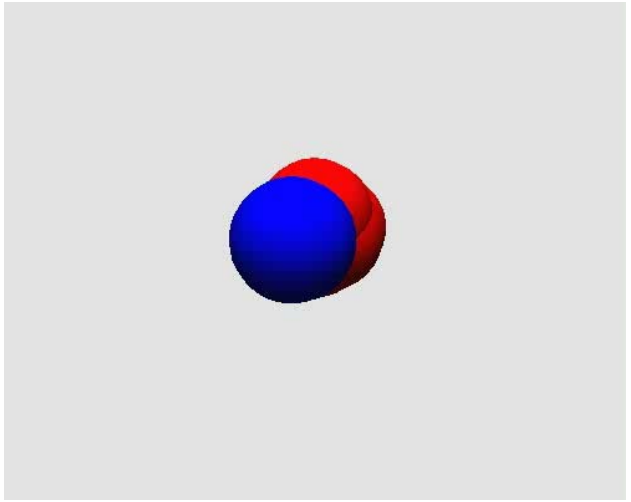
Cellular motility

Adaptive mesh

Neurulation

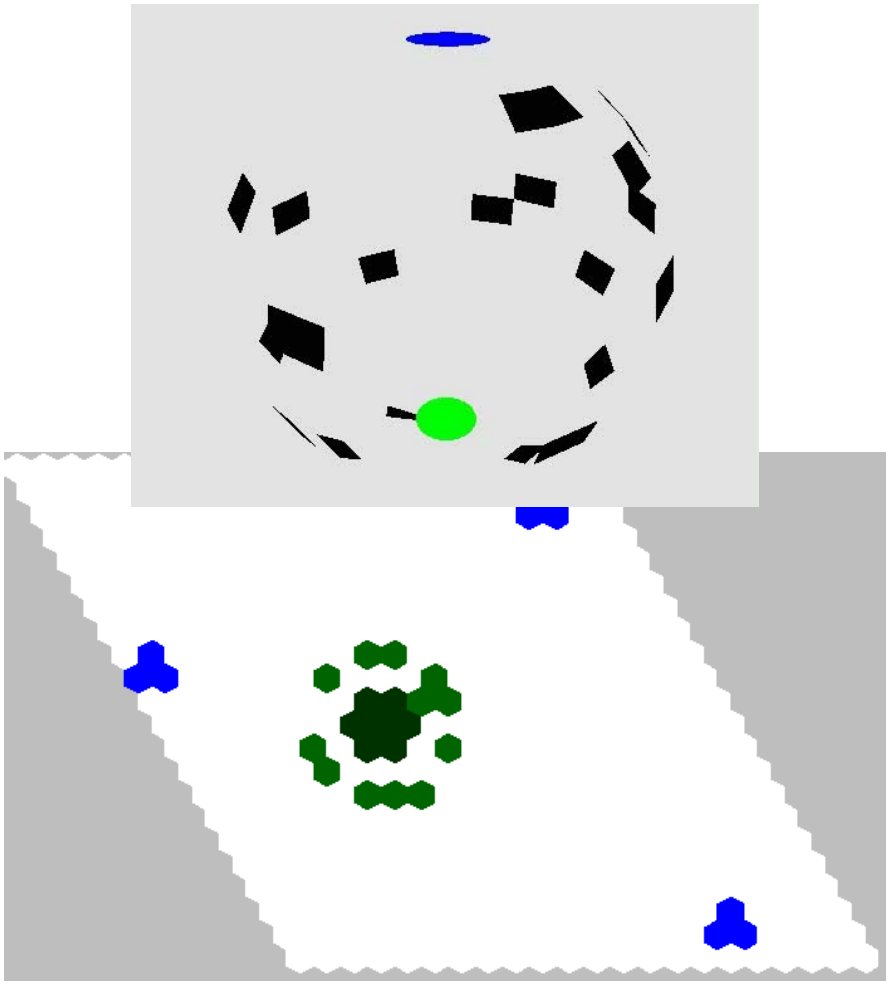
Modification of topology





Tumor growth

Programming a population



Ants foraging

One transformation for any topologies: *polytypism*

Flocking birds

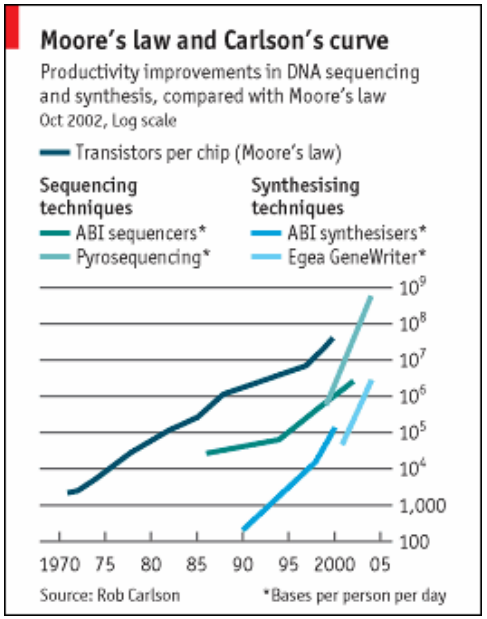
- No leader
- 3 evolution rules
- A coherent global behavior

Outline

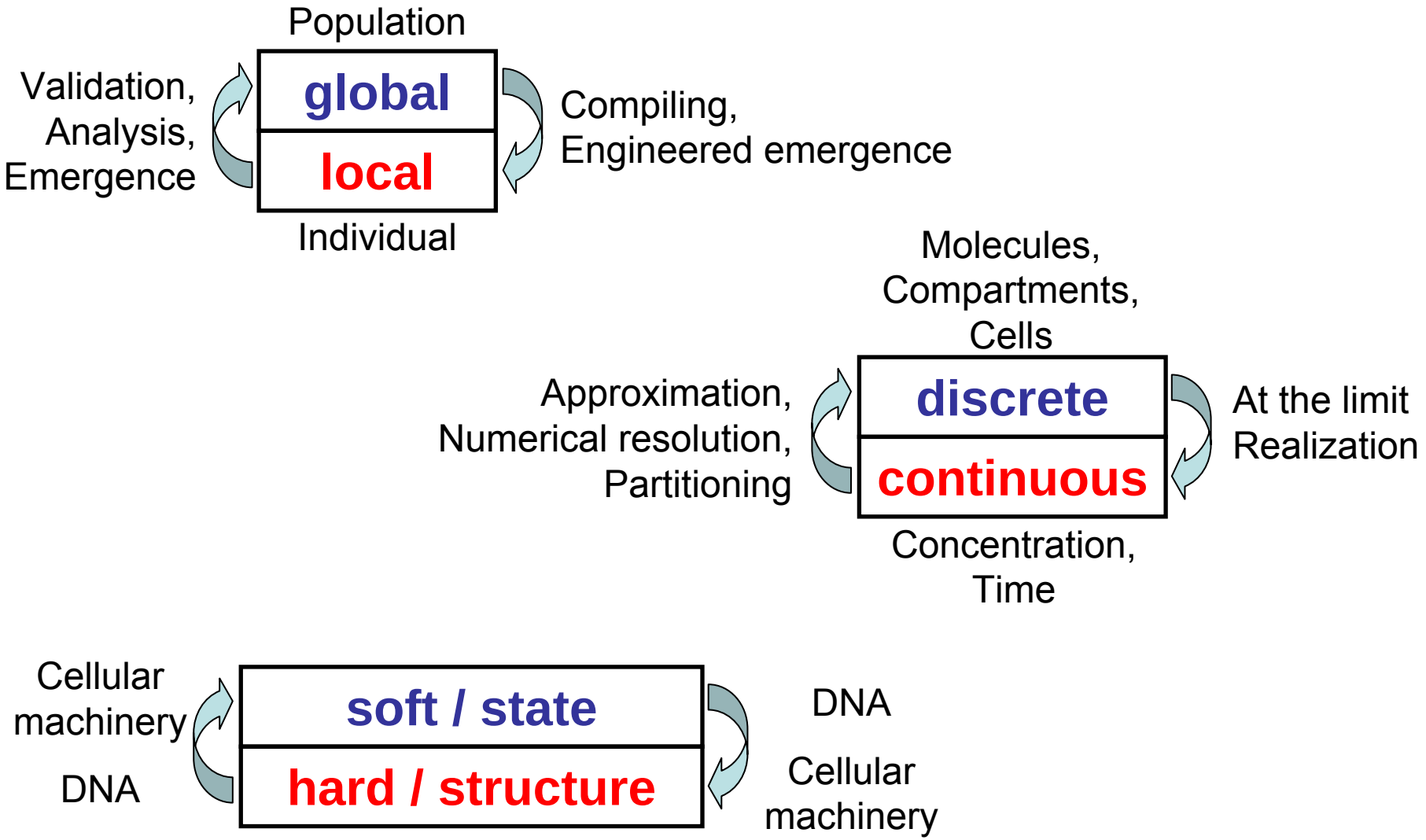
- MGS Project
  - ✓ Topological collections
  - ✓ Transformations
  - ✓ Examples of applications
- Spatial programming of a biosynthetic medium
  - ✓ Biosynthetic: a wetware
  - ✓ Algebraic programming
  - ✓ Discrete differential calculus
- Conclusion & perspectives

Biosynthetic

- Spatial programming of a biosynthetic medium
- Biological entities exhibit a form of reactive computation
  - ✓ Perceive their environment
  - ✓ Process information
  - ✓ Modify their state and act on the environment
- Synthetic biological systems
  - ✓ **Available technology**
    - Design on a large scale of biological systems
    - *Moore law*-like progression (Carlson's curve)
  - ✓ **Applications**
    - ***Biological computation*** (handling chemical compounds, nanobiology, biosensors, filters, …)
    - Dragging out design principles to develop ***autonomic computing systems*** (self-* properties)
    - Understanding life by ***building*** biological systems



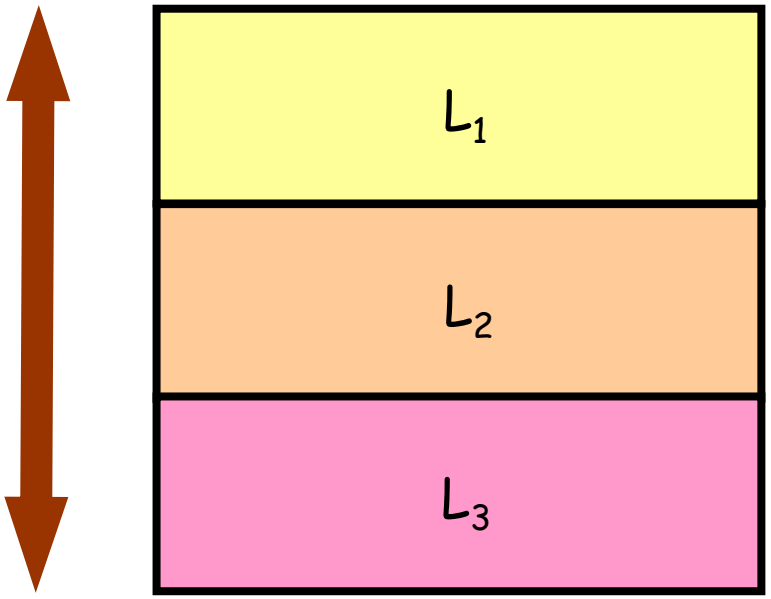
Three challenges



Short and mid term goals

- Using results from spatial programming, amorphous computing and synthetic biology
- Compiling tower
  - ✓ Three different ***levels of expression*** (L_i)
  - ✓ Analysis and ***compiling*** tools ($L_i \Leftrightarrow L_{i+1}$)

High level expression (space)



Hardware (amorphous medium)

Spatial Programming

Amorphous computing
(global/local)

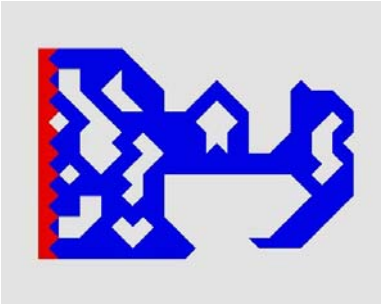
Control theory
dynamical systems
(digital/analogical)

Genetic engineering, *wetware*
(software/hardware)

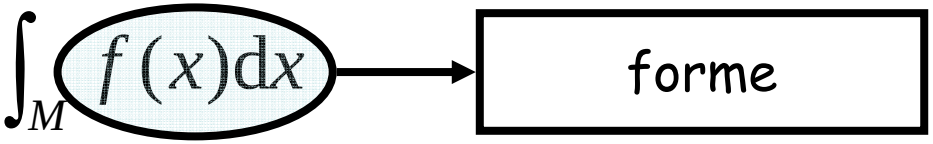
Synthetic biology

Global/local compiling

Designing L_1



- ✓ Inspiration: developmental biology
  - Morphogenetic gradient
  - Wave propagation
  - Regulation, diffusion of chemicals
- ✓ Design patterns from the differential calculus point of view
  - Differential form



  - Data structure as a topological space
    - Differential form = homomorphism of data structures
    - Algebra based on differential calculus operators
  - Advantages
    - Concise and usual expression (differential equation, discrete physics)
    - Bird-Merteens algebra (intentional handling of lists)

Global/local compiling

Differential calculus and compiling

- ✓ Differential form
  - Local object
  - Applied on each point of domain (integration)
  - Computation of a global result

$$\int_M f(x)\mathrm{d}x$$

- ✓ Mathematical corpus
  - Boundary operators
  - Stockes theorem

$$\int_M f(x)\mathrm{d}x = \int_M \mathrm{d}F(x) = \oint_{\partial M} F(x)$$

  - Physical operators: gradient, divergence, laplacian, curl, …
  - Solving differential equations

- ✓ Compiling

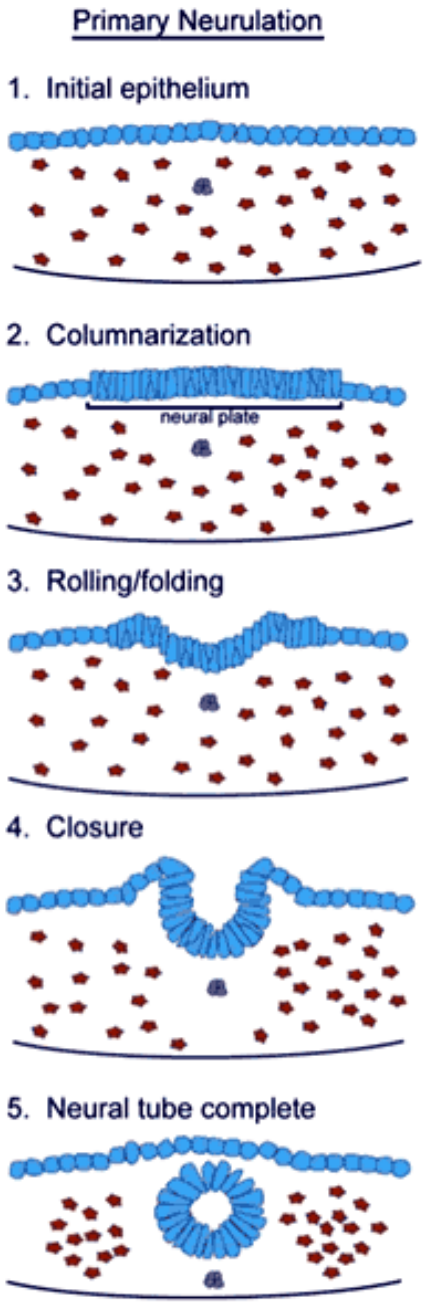
  Differentiation of *global* behavior into *local* behaviors

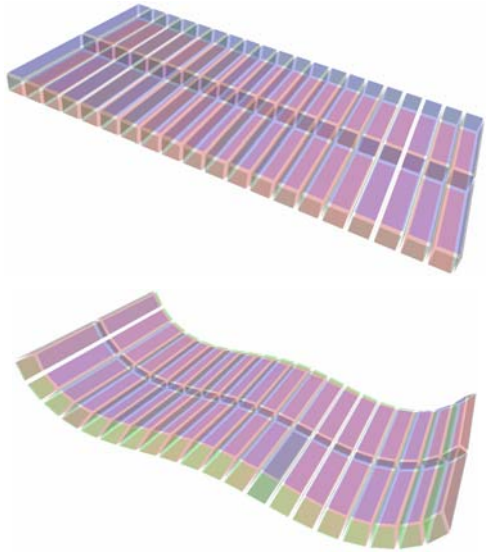
Outline

- MGS Project
  - ✓ Topological collections
  - ✓ Transformations
  - ✓ Examples of applications
- Spatial programming of a biosynthetic medium
  - ✓ Biosynthetic: a wetware
  - ✓ Algebraic programming
  - ✓ Discrete differential calculus
- Conclusion

Conclusion

- Spatially organized systems
  - ✓ Dynamical systems with a dynamical structure
  - ✓ Composed of autonomous elements
- MGS: a topological approach of spatial computing
  - ✓ Topological collection: spatial organization
  - ✓ Transformation: local behaviors and interactions
- Programming a biosynthetic medium
  - ✓ A tower of languages/levels of expressions
  - ✓ Programming at a global point of view
    - Developmental biology inspiration
    - Use of differential operators
  - ✓ An amorphous biological computing medium
    - MIT Biobricks, biological parts
    - iGEM challenge (http://biosynthetique.free.fr/)





Questions?

http://mgs.ibisc.univ-evry.fr