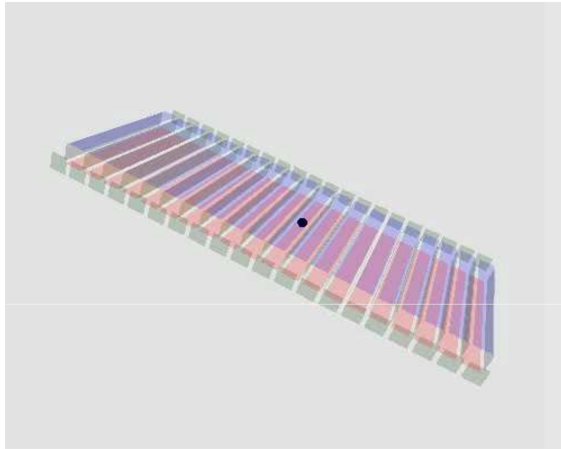
Domain-specific Language for the Modeling of Dynamical Systems with a Dynamical Structure



Olivier MICHEL
LACL, Univ. Paris Est

http://lacl.univ-paris12.fr/Labo/Membres/oliviermichel.html

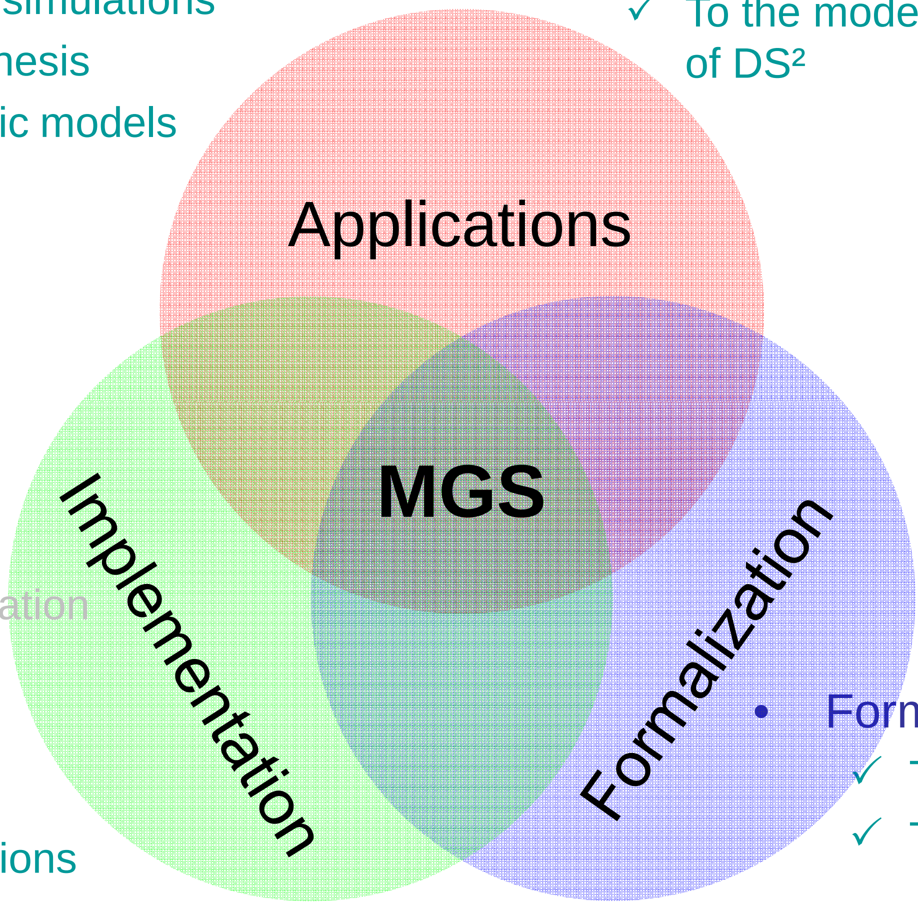
&

http://mgs.spatial-computing.org

MGS Project

- Examples
  - ✓ Topological modifications
  - ✓ Numerical simulations
  - ✓ Morphogenesis
  - ✓ Probabilistic models

- Motivations: *dsl* dedicated
  - ✓ To space and fields representation
  - ✓ To the modeling and the simulation of DS²



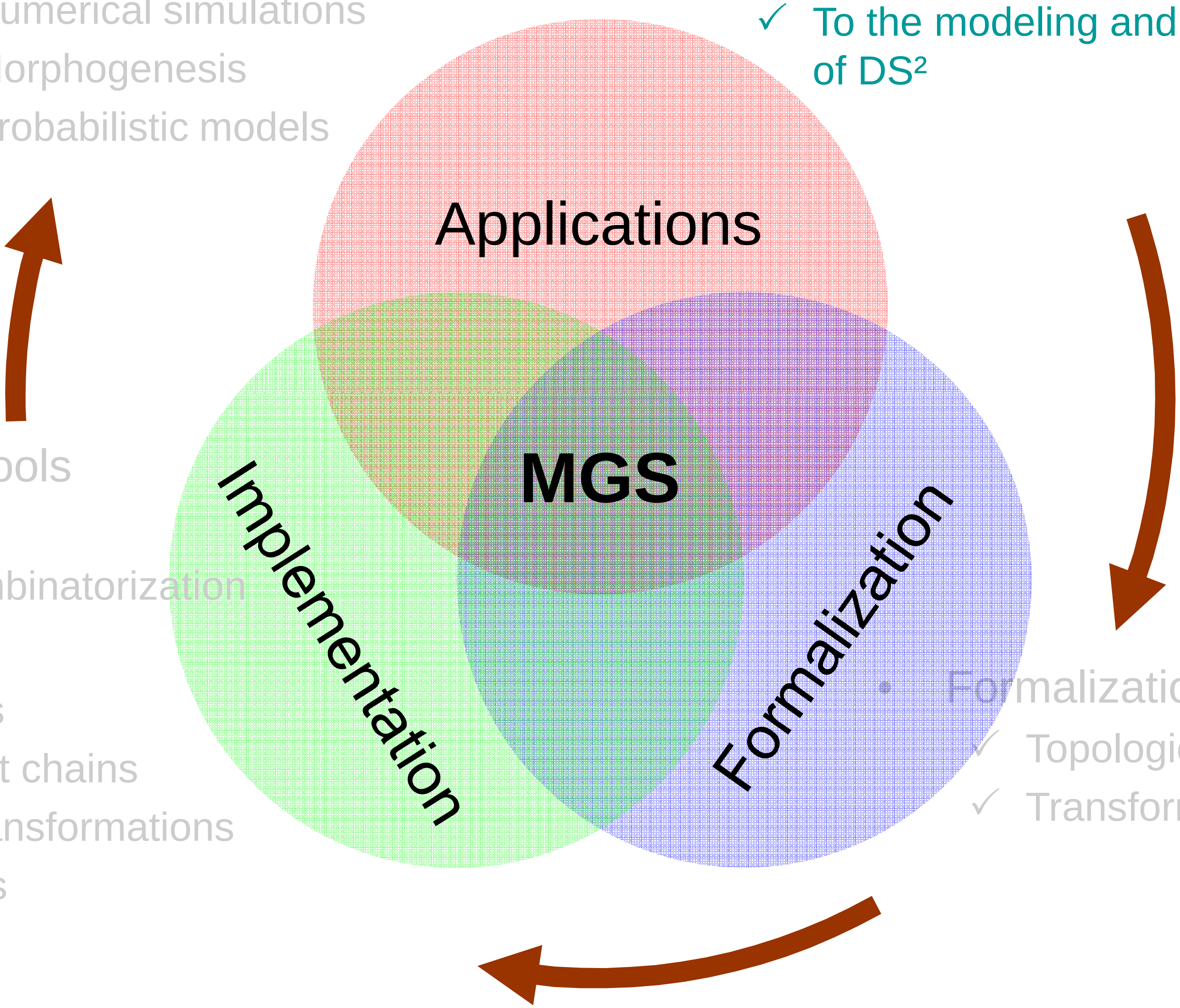
- Compiler/tools
  - ✓ Typing
  - ✓ SK-combinatorization
- Toplevel
  - ✓ G-maps
  - ✓ Abstract chains
  - ✓ Path transformations
  - ✓ Patches
  - ✓ …

- Formalization
  - ✓ Topological collections
  - ✓ Transformations

MGS Project

- Examples
  - ✓ Topological modifications
  - ✓ Numerical simulations
  - ✓ Morphogenesis
  - ✓ Probabilistic models

- Motivations: *dsl* dedicated
  - ✓ To space and fields representation
  - ✓ To the modeling and the simulation of DS²



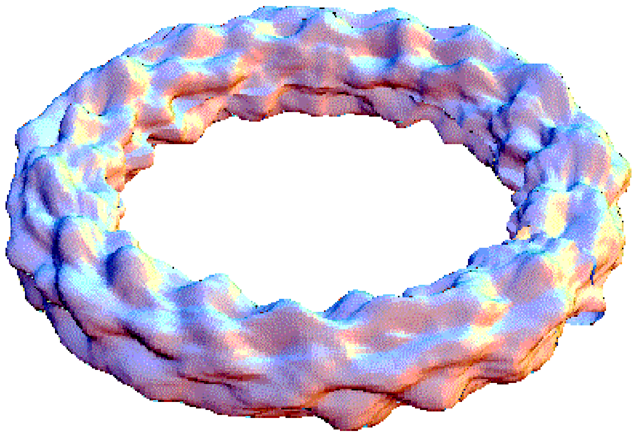
- Compiler/tools
  - ✓ Typing
  - ✓ SK-combinatorization
- Toplevel
  - ✓ G-maps
  - ✓ Abstract chains
  - ✓ Path transformations
  - ✓ Patches
  - ✓ …

- Formalization
  - ✓ Topological collections
  - ✓ Transformations

Space and Fields Representation

Our goal:

- ✓ Enhancing programming languages with spatial relationships
- ✓ Making space explicit in the definition
- ✓ Having explicit spatial operators



```
gbf torus =
   < a, b; 180 a = 0, 60 b = 0
>
x => NeighborsFold(f, e, x)
```

Data-structure = Field on Topological Space

Computation = Movement in this Space

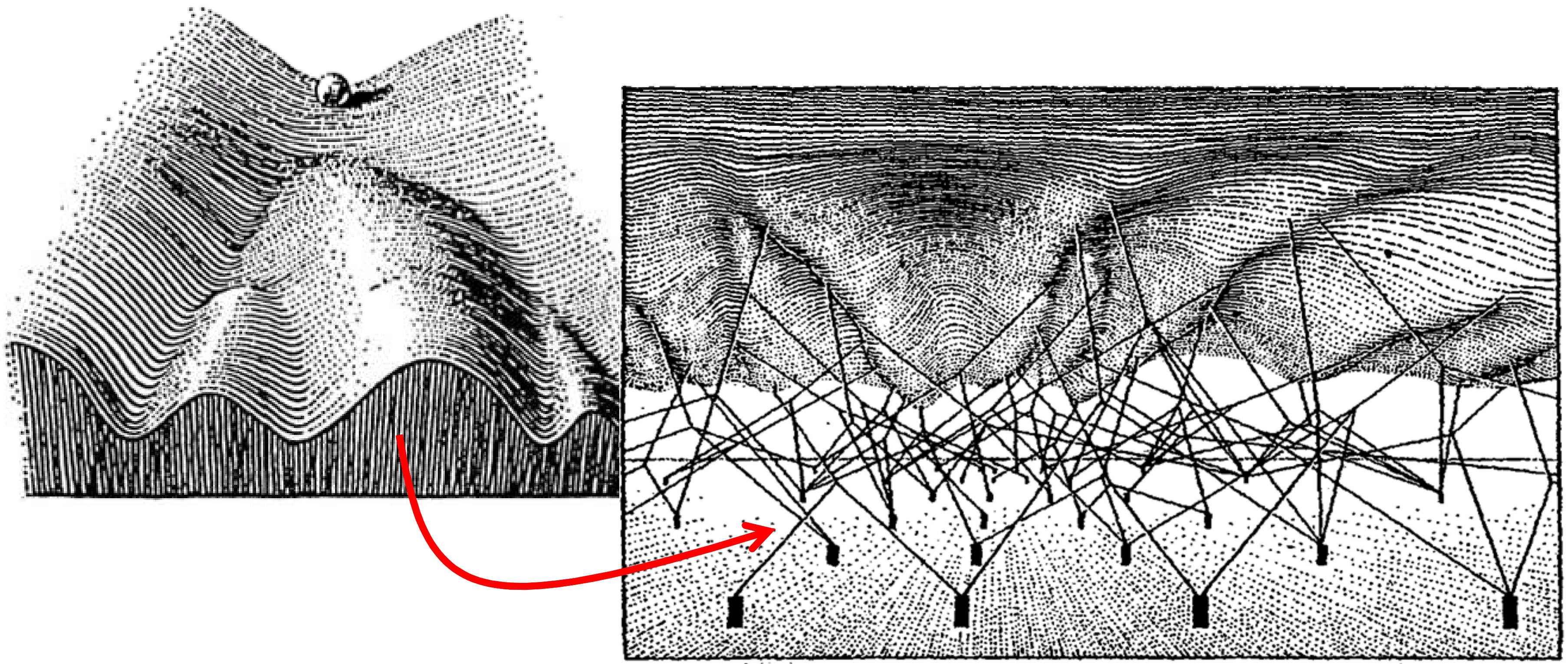
⇒ Spatial Computing

Motivations – DS^2

- Dynamical System with a Dynamical Structure
  - ✓ Complex systems
  - ✓ Whose structure evolves over time
    - The structure constrains the behavior of the system that modifies the structure
    - The phase space cannot be defined ***a priori***
    - The evolution function cannot be specified **globally**

[Giavitto & Michel – *Technical Report* - 2001]
[Giavitto & Michel - *Fundamenta Informaticae* - 2002]

…



Epigenetic landscape and chreods by Conrad Hall Waddington

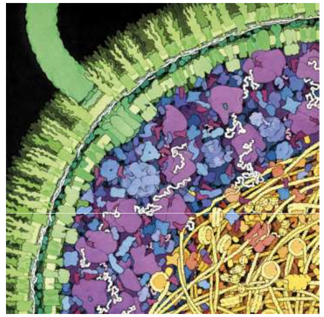
Motivations – DS^2

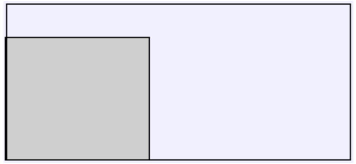
- Dynamical System with a Dynamical Structure
  - ✓ Complex systems
  - ✓ Whose structure evolves over time
    - The structure constrains the behavior of the system that modifies the structure
    - The phase space cannot be defined ***a priori***
    - The evolution function cannot be specified **globally**

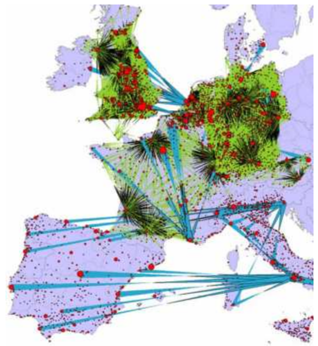
[Giavitto & Michel – *Technical Report* - 2001]
[Giavitto & Michel - *Fundamenta Informaticae* - 2002]

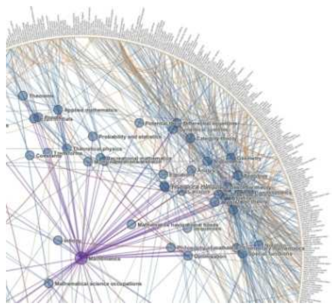
…

- Examples
  - ✓ Biology
    - Molecular biology
      (regulatory, metabolic, signalization networks)
    - Developmental biology
      (plant, embryo…)
  - ✓ Physics
    - Soft matter mechanics,
      multi-scale systems
  - ✓ Urbanism
    - City growth,
      road traffic control, …
  - ✓ Computer science
    - Internet, sensor network,
      reconfigurable nanobots,
      knowledge network, …


© David S. Goodsell 1999


© Tecplot

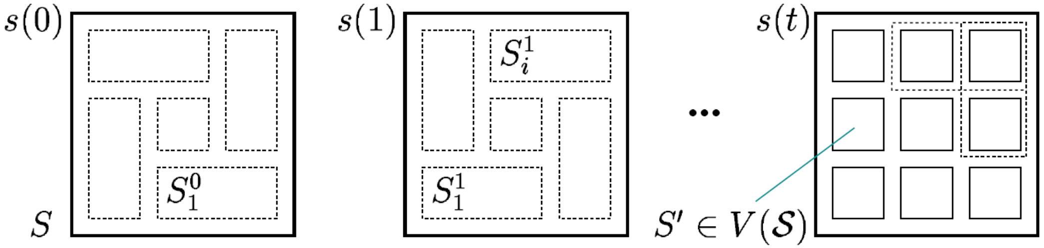

© L. Sanders - EUROSIM


© C. Harrison - Clusterball project

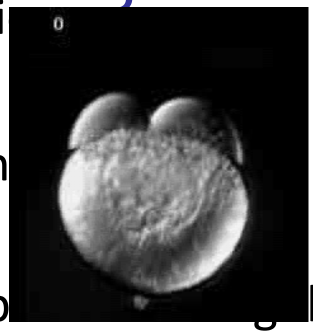
Motivations – DS²

- Interactions in DS²
  - ✓ Sub-systems evolution in parallel
    - *s(t)* : state of the system at time *t*
    - S_i^t : *i*ème sub-system where an interaction occurs
  - ✓ The successive decompositions S_1^t , S_2^t , S_3^t , … capture
    - The atomic parts of the systems: **entities**
    - The organization of these parts: the **space of interactions**
    - The local evolution rules

- What kind of language to express such interacti… ?



…ons of en…



The global behavior arises from local interactions b… …bors

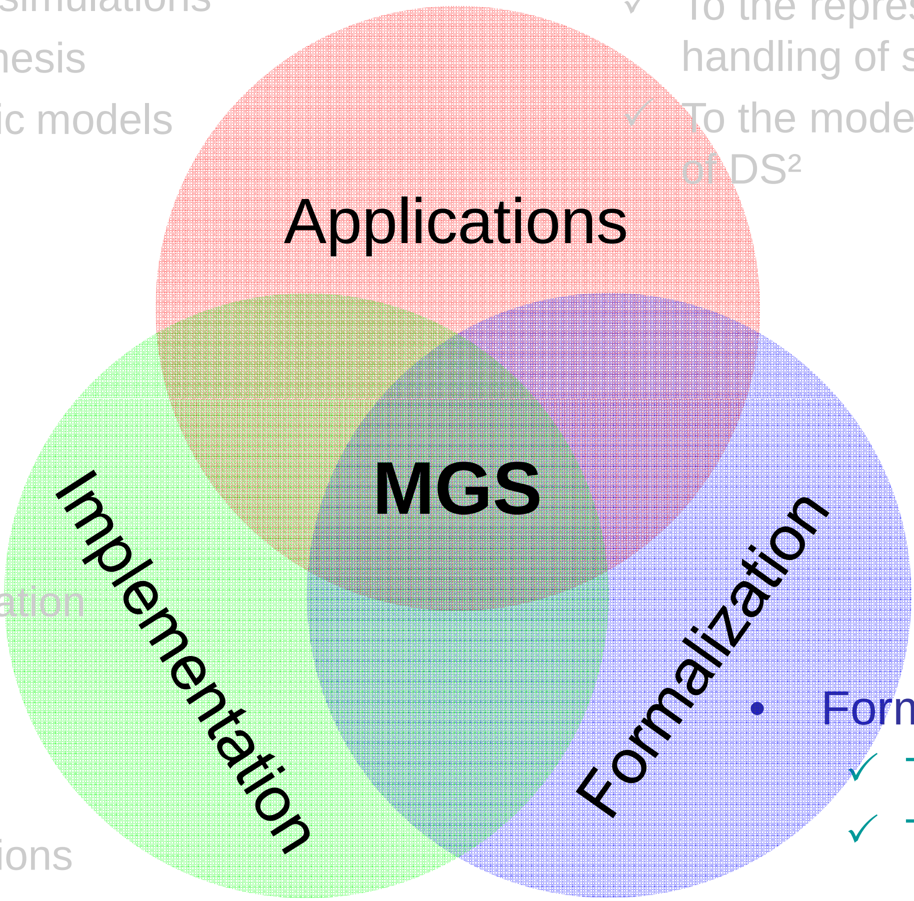
  - ✓ **Declarative**

    Close to mathematical model to ease the use of formal tools

MGS Project

- Examples
  - ✓ Topological modifications
  - ✓ Numerical simulations
  - ✓ Morphogenesis
  - ✓ Probabilistic models

- Motivations: domain specific language dedicated
  - ✓ To the representation and the handling of space
  - ✓ To the modeling and the simulation of DS^2



- Compiler
  - ✓ Typing
  - ✓ SK-combinatorization
- Toplevel
  - ✓ G-maps
  - ✓ Abstract chains
  - ✓ Path transformations
  - ✓ Patches
  - ✓ …

- Formalization
  - ✓ Topological collections
  - ✓ Transformations

Rewriting Systems

Complex systems ⟺ Rewriting techniques

Modeling	Definition
State (space)	Data structure
Hierarchical organizations	Formal trees (terms)
Evolution function	Rewriting system
interaction → result	$\alpha \rightarrow \beta$ α: pattern, β: expression
Local evolution laws	Set of rules
Simulation	**Application**
Trajectories	Derivations
Time managing	Rule application strategies
Discrete, event based: synchronous/asynchronous/…	Maximal parallel/sequential/ stochastic/…

Rewriting Systems

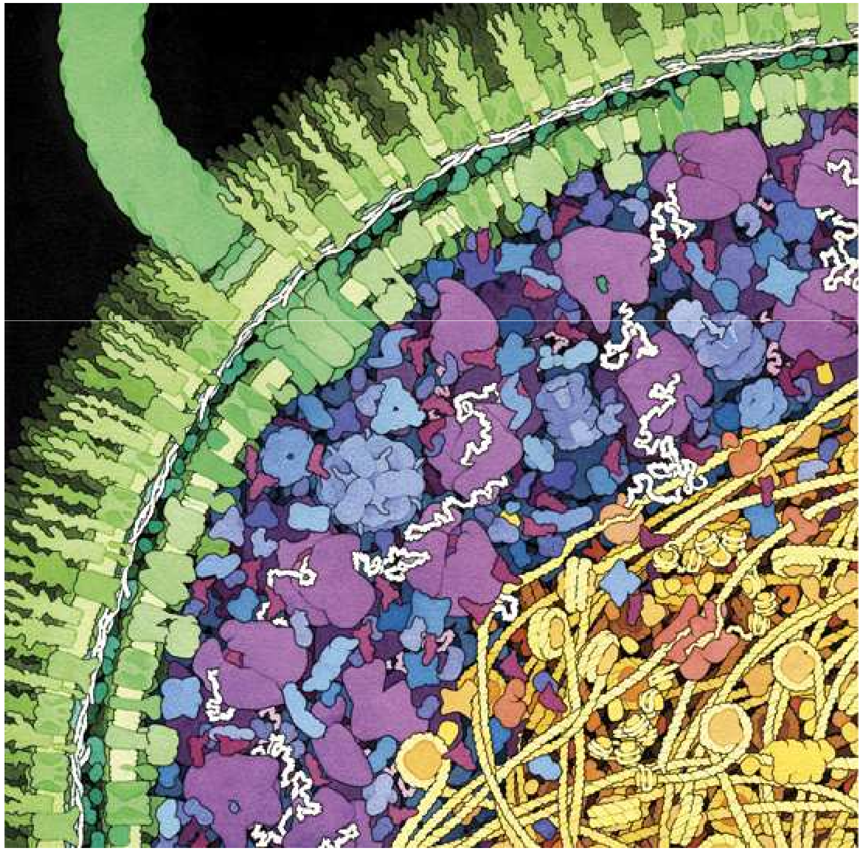
Complex systems ⟺ Rewriting techniques

Modeling	Definition
State (space)	**Data structure**
Hierarchical organizations	Formal trees (terms)
Arbitrary organizations	*What kind of data structure?*
Evolution function	**Rewriting system**
interaction → result	α → β α: pattern, β: expression
Local evolution laws	Set of rules

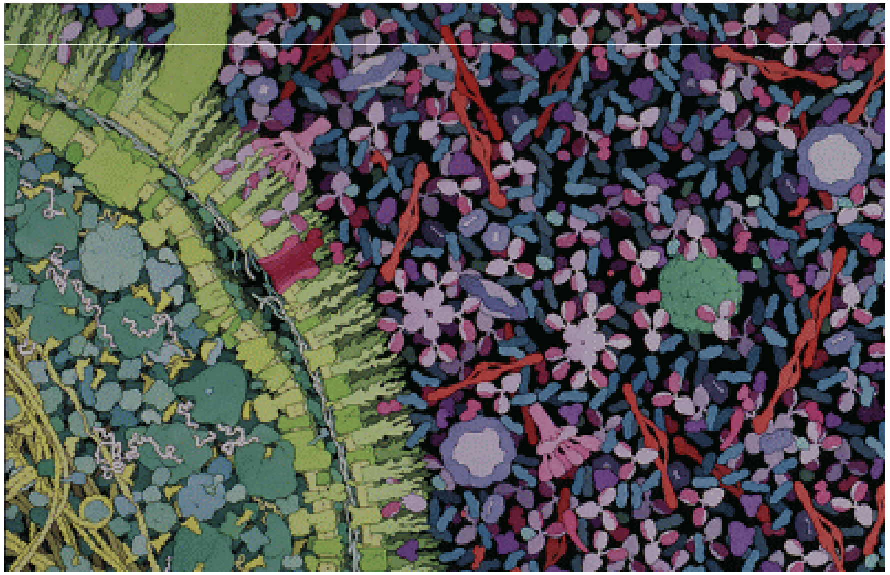
Simulation	Application
Trajectories	**Derivations**
Time managing	**Rule application strategies**
Discrete, event based: synchronous/asynchronous/…	Maximal parallel/sequential/ stochastic/…

Complex Spatial Structures

- Can really a cell be represented by a (formal) tree?



© David S. Goodsell 1999
http://mgl.scripps.edu/people/goodselll



© David S. Goodsell 1999

Rewriting Systems

Complex systems ⟺ Rewriting techniques

Modeling	Definition
State (space)	**Data structure**
Hierarchical organizations	Formal trees (terms)
Arbitrary organizations	*Topological collections*
Evolution function	**Rewriting system**
interaction → result	α → β α: pattern, β: expression
Local evolution laws	Set of rules, *Transformation*

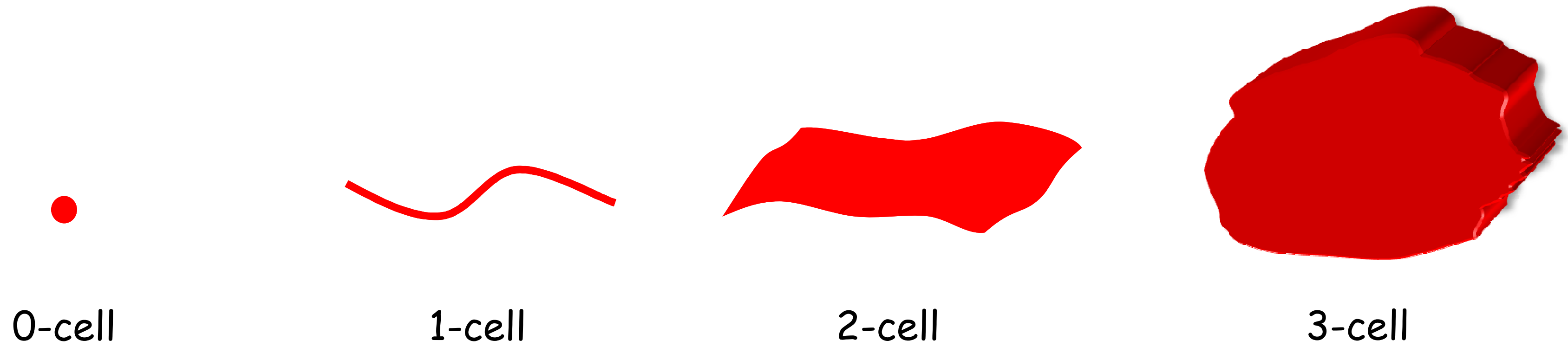
Simulation	Application
Trajectories	**Derivations**
Time managing	**Rule application strategies**
Discrete, event based: synchronous/asynchronous/…	Maximal parallel/sequential/ stochastic/…

Topological Collection

The Topological collections

✓ **Structure**:

- A collection of *topological cells*

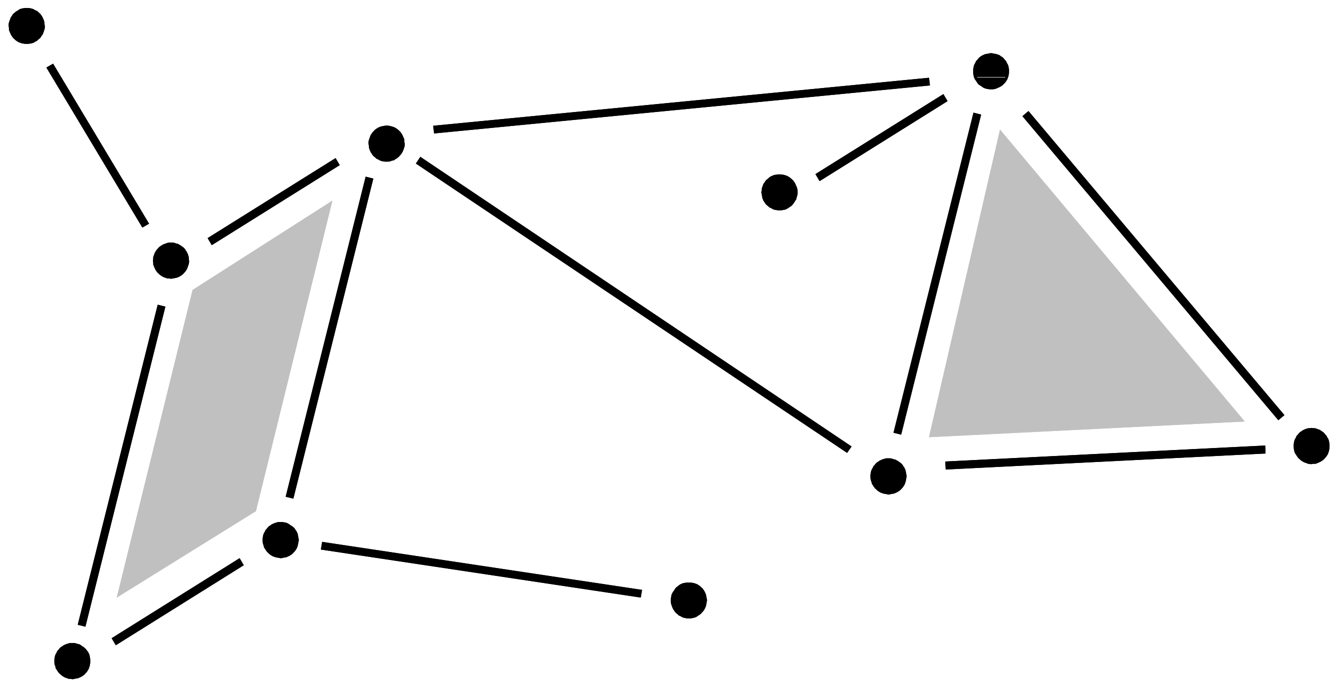


Topological Collection

The Topological collections

✓ **Structure**:

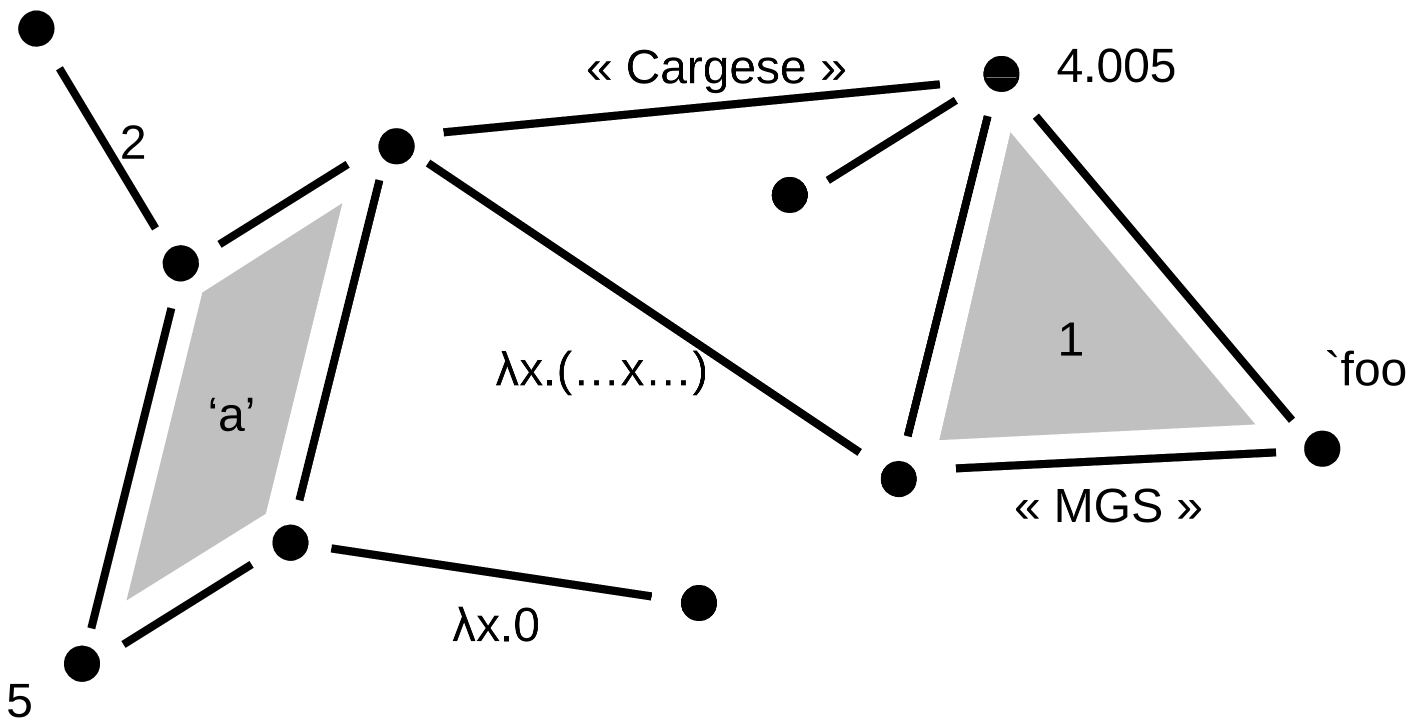
- A collection of topological cells
- An *incidence relationship*



Topological Collection

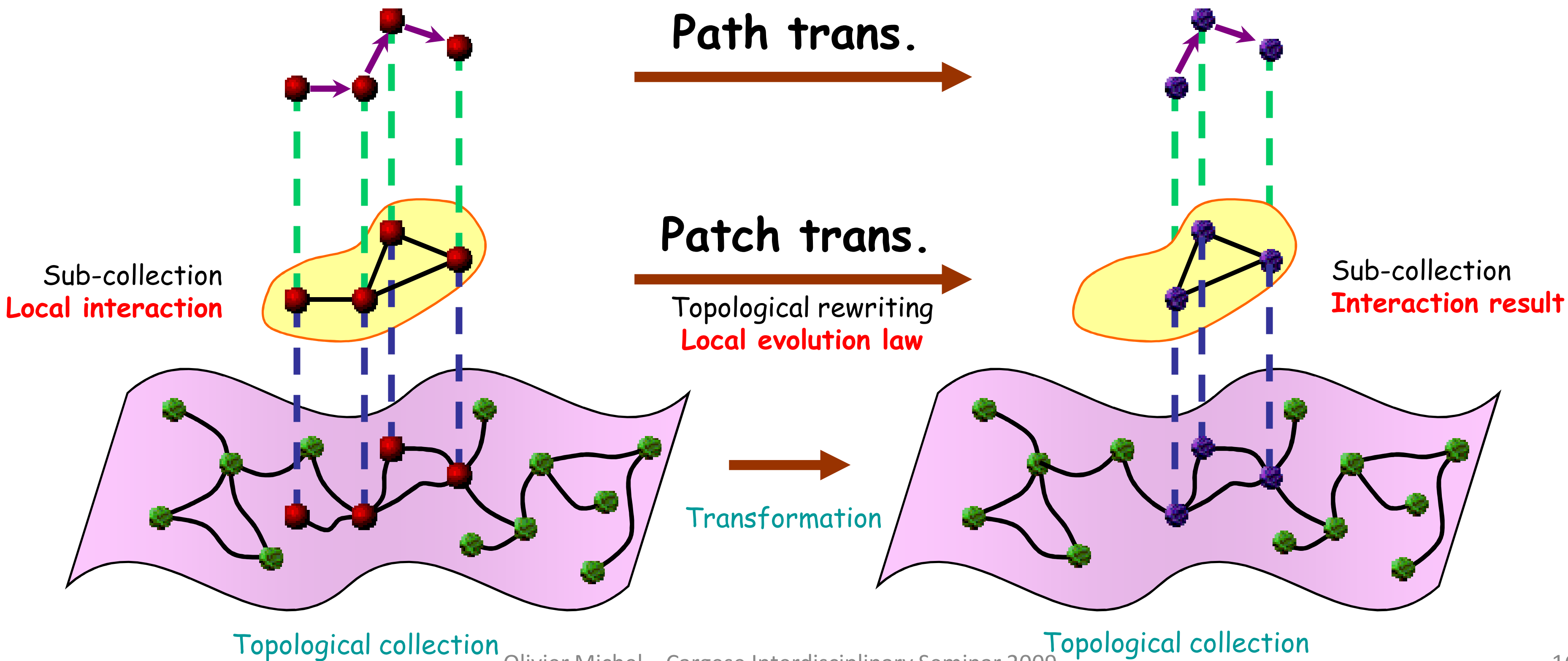
The Topological collections

- ✓ **Structure**:
  - A collection of topological cells
  - An incidence relationship
- ✓ **Data**: *associate a value with each cell*



Transformation

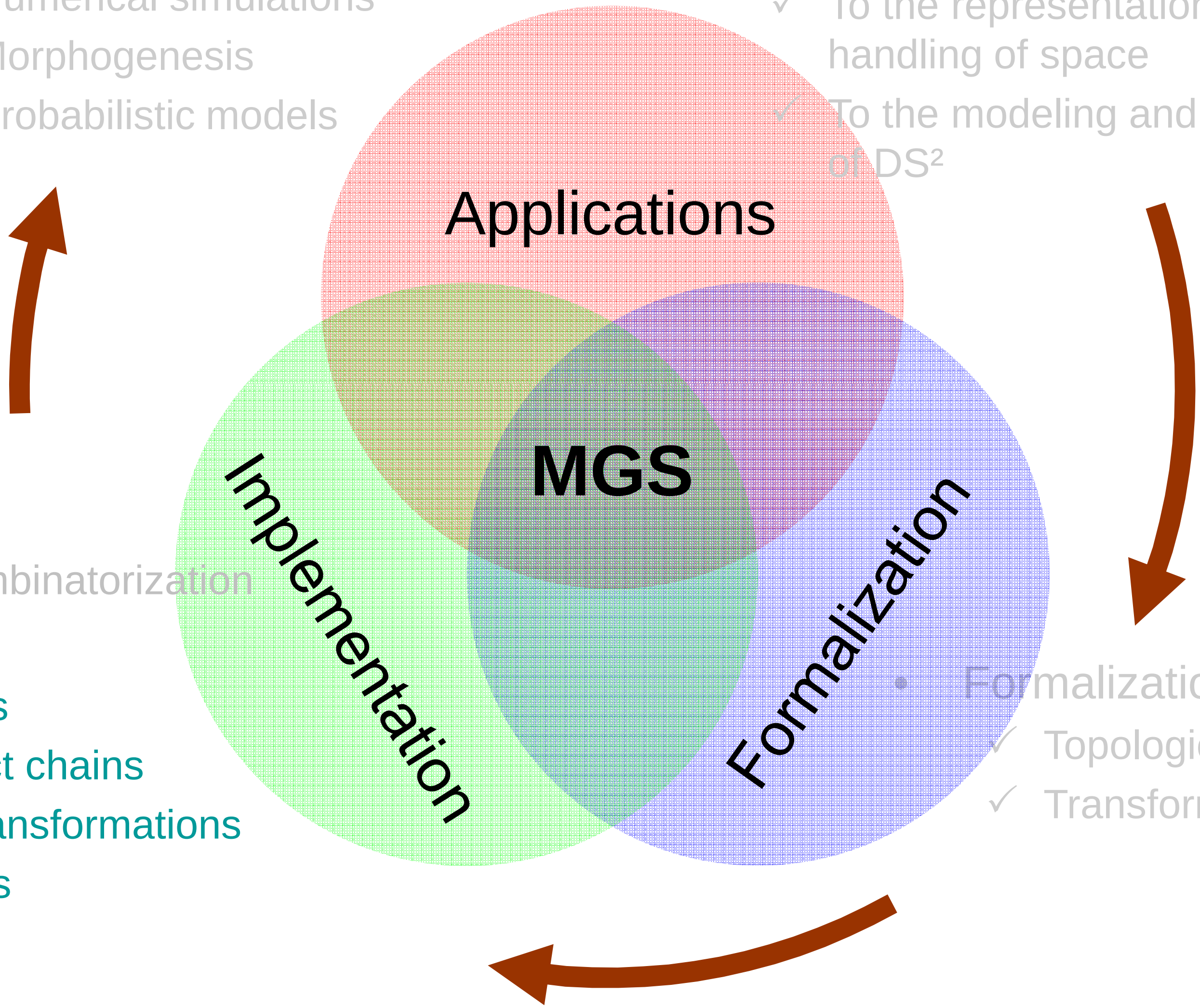
- ✓ **Functions** on collections
- ✓ Defining a **rewriting relationship**
  Local rewriting of collections: *topological rewriting*
- ✓ Function **defined by case**
  Each case (pattern) *matches* a sub-collection



MGS Project

- Examples
  - ✓ Topological modifications
  - ✓ Numerical simulations
  - ✓ Morphogenesis
  - ✓ Probabilistic models

- Motivations: domain specific language dedicated
  - ✓ To the representation and the handling of space
  - ✓ To the modeling and the simulation of DS²



- Compiler
  - ✓ Typing
  - ✓ SK-combinatorization
- Toplevel
  - ✓ G-maps
  - ✓ Abstract chains
  - ✓ Path transformations
  - ✓ Patches
  - ✓ …

- Formalization
  - ✓ Topological collections
  - ✓ Transformations

The MGS Language

- Implementing Topological Rewriting in a Language

- Data structures
  - ✓ Topological collections
    the *only* data structure available in MGS
  - ✓ Standard data structures as collections
    sequence, generalized array, (multi-)set, arbitrary graph, Delaunay triangulation, G-map, …

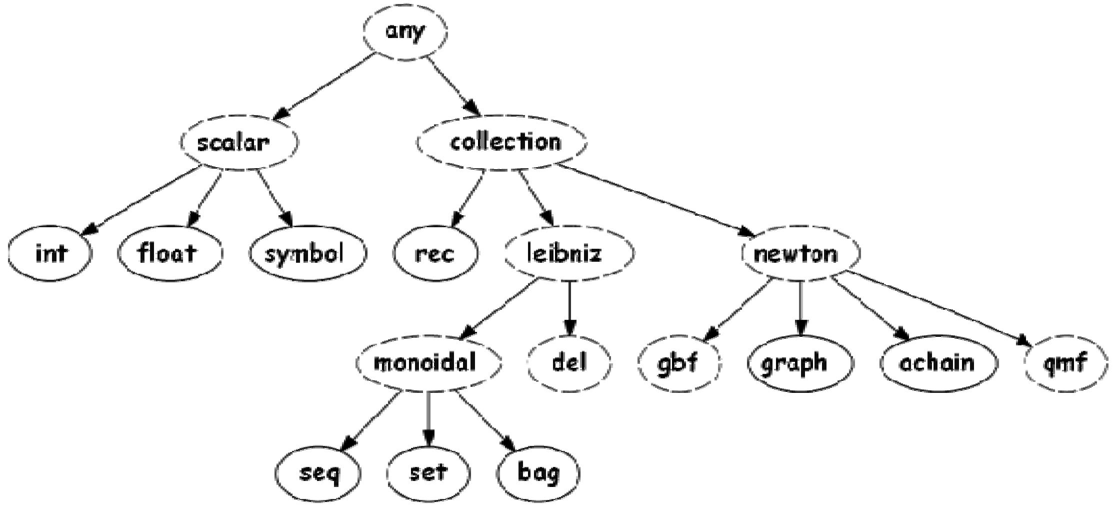
- Control structures
  - ✓ ML-like functional language (higher-order language)
    Functions definition and application
  - ✓ Transformations
    - Two ***pattern*** languages
      path and patch
    - Many ***rule application strategies***
      maximal parallel, asynchronous, stochastic, Gillespie' SSA based, …
    - Transformation ***evaluation***

$$\mathbf{Trans}\ T = \{$$
$$rule_1 : pattern_1 \Rightarrow expression_1$$
$$rule_2 : pattern_2 \Rightarrow expression_2$$
$$\ldots$$
$$rule_n : pattern_n \Rightarrow expression_n$$
$$\}$$

$$T[\text{strategy} = \ldots, \ldots](c)$$

MGS Implemented

- Interpreter
  - ✓ MGS: 50k lines of ML/C/libraries, 3rd version
  - ✓ Source freely available
  - ✓ **`http://mgs.spatial-computing.org`**



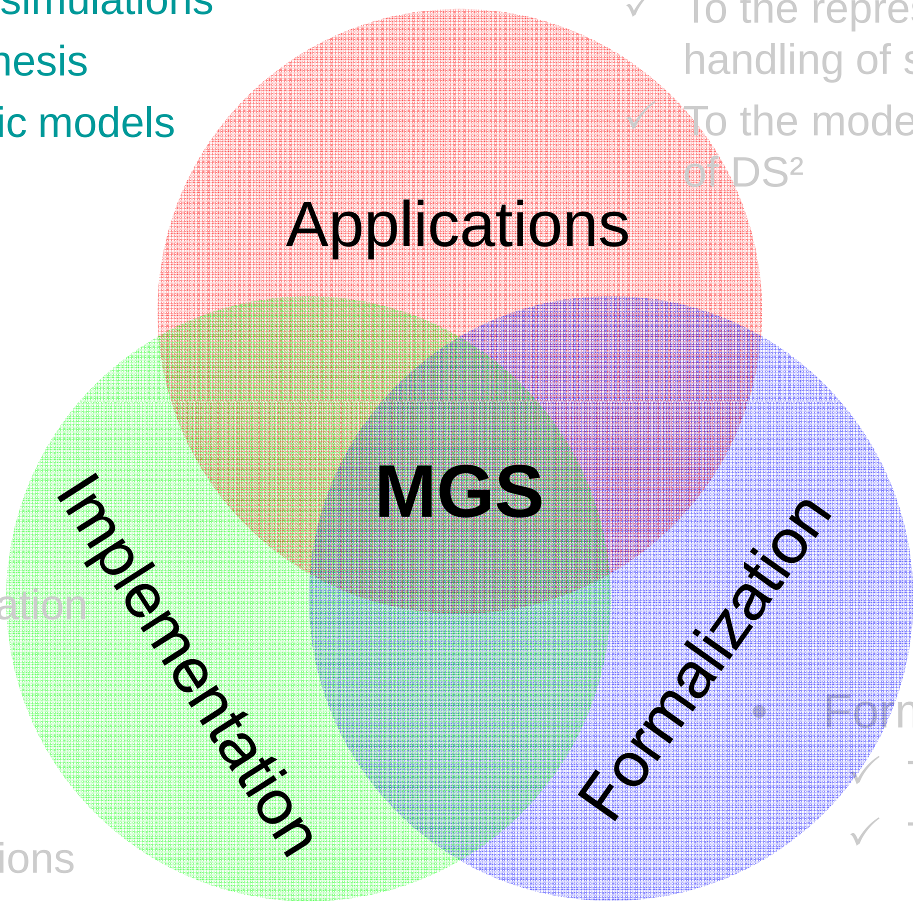
- Companion softwares
  - ✓ Imoview, PatchGen, GBview…

- Examples and applications

MGS Project

- Examples
  - ✓ Topological modifications
  - ✓ Numerical simulations
  - ✓ Morphogenesis
  - ✓ Probabilistic models

- Motivations: domain specific language dedicated
  - ✓ To the representation and the handling of space
  - ✓ To the modeling and the simulation of DS²



- Compiler
  - ✓ Typing
  - ✓ SK-combinatorization
- Toplevel
  - ✓ G-maps
  - ✓ Abstract chains
  - ✓ Path transformations
  - ✓ Patches
  - ✓ …

- Formalization
  - ✓ Topological collections
  - ✓ Transformations

Some Examples

matching pattern

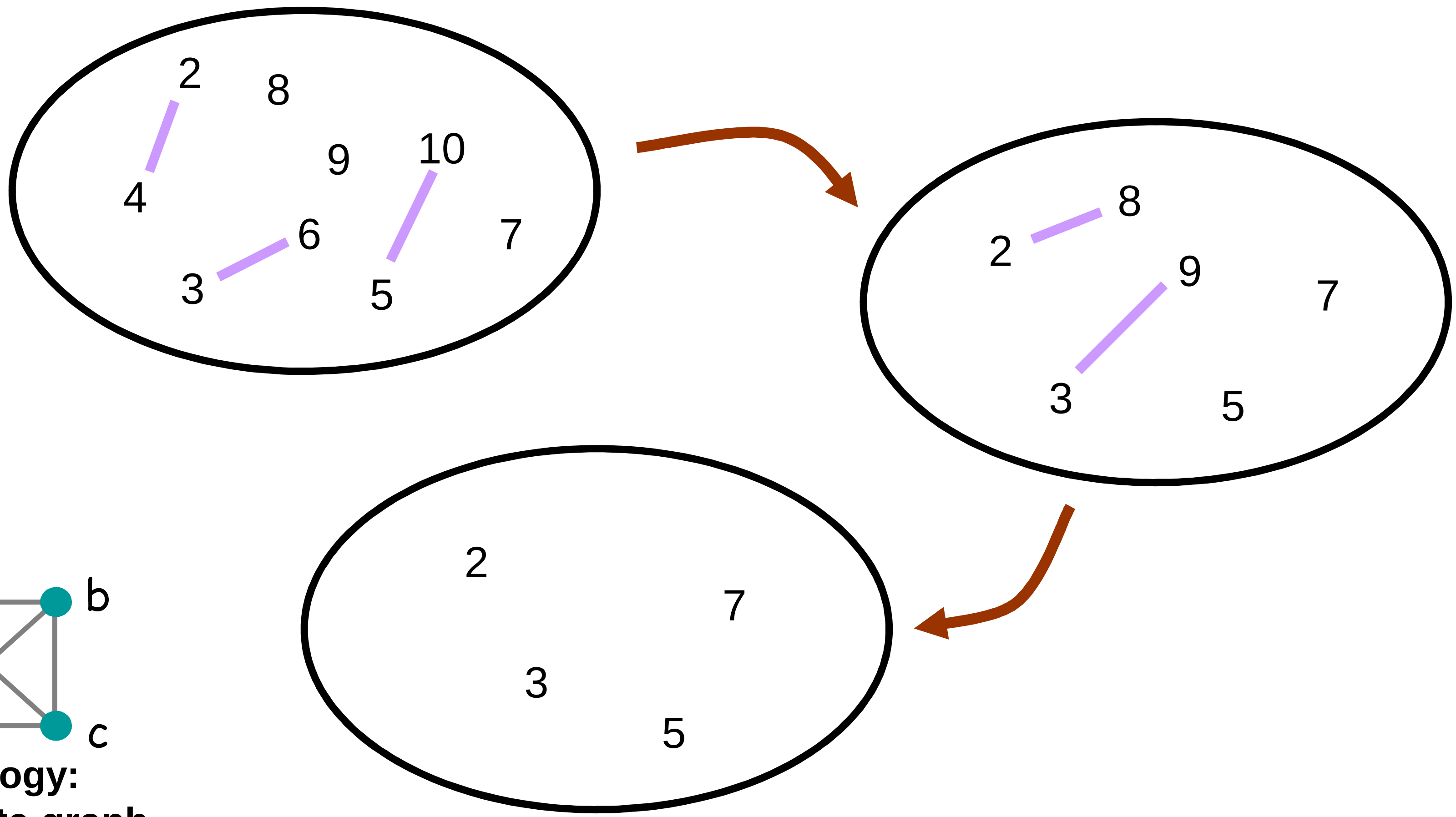
```
x, y / x div y => x
```

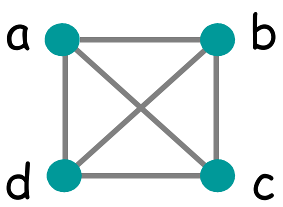
Find x neighbor of y such that y is divided by x
Replace (x, y) by x

Some Examples

- Multi-set rewriting (chemical calculus and P systems): prime numbers computation

```
trans { x, y / x div y => x }
```





Topology:
a complete graph

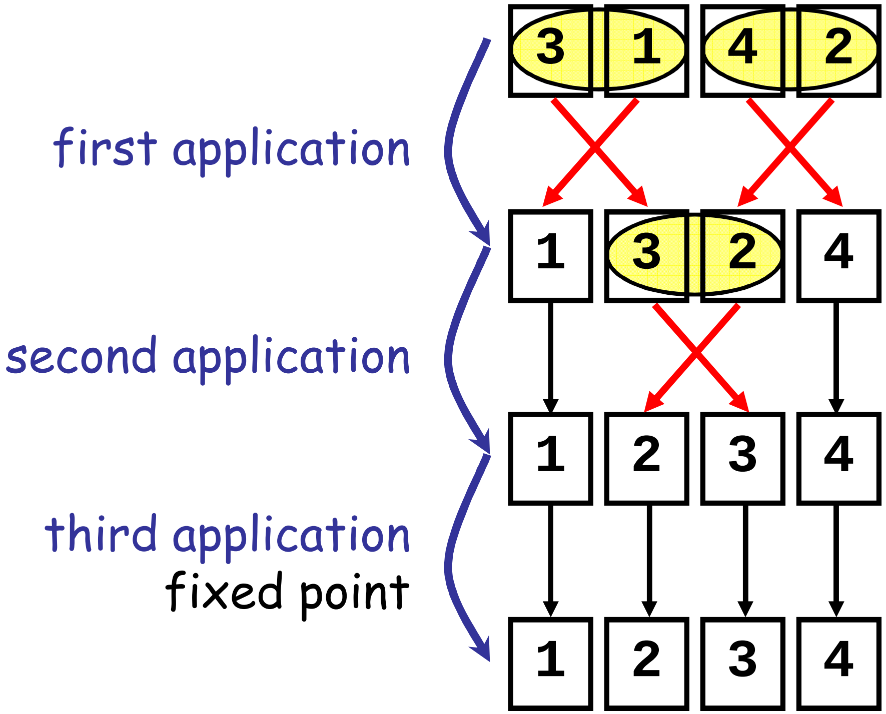
Some Examples

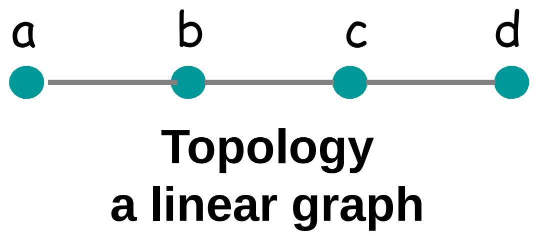
- Multi-set rewriting (chemical calculus and P systems): prime numbers computation

```
trans { x, y / x div y => x }
```

- Sequence rewriting (L system): bubble sort

```
trans { x, y / x > y => y, x }
```





Some Examples

- Multi-set rewriting (chemical calculus and P systems): prime numbers computation

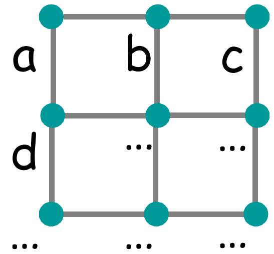
```
trans { x, y / x div y => x }
```

- Sequence rewriting (L system): bubble sort

```
trans { x, y / x > y => y, x }
```

- Array rewriting (cellular automaton): bead sort

```
trans { `Bead |south> `Empty => `Empty, `Bead }
```



Topology:
a regular graph ≡ GBF

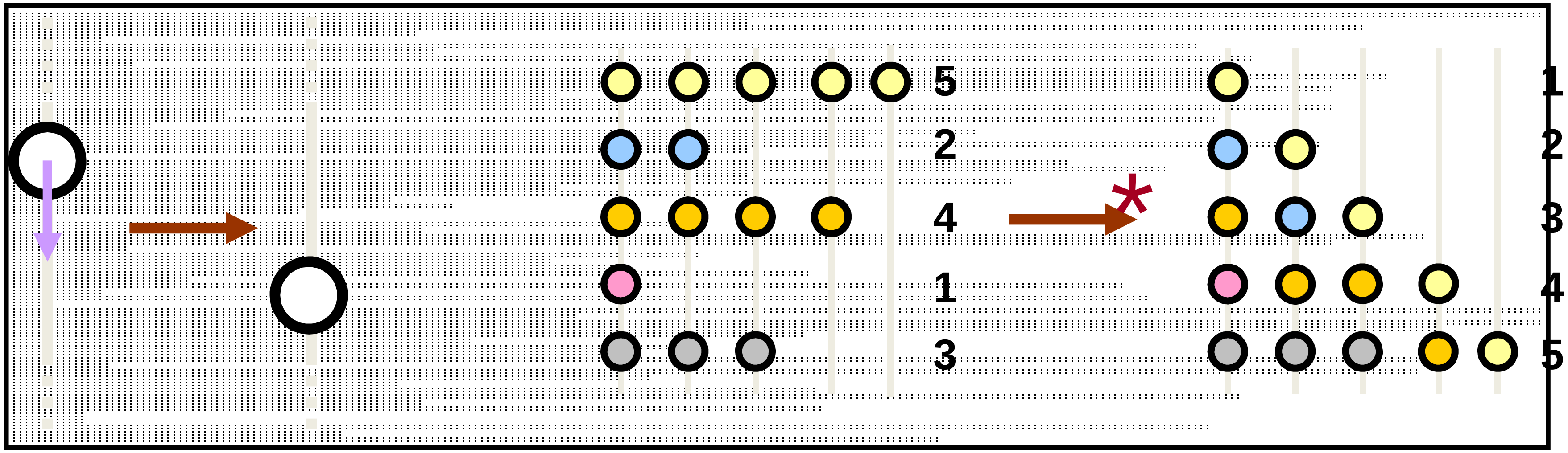
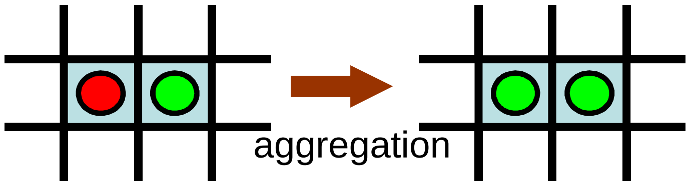
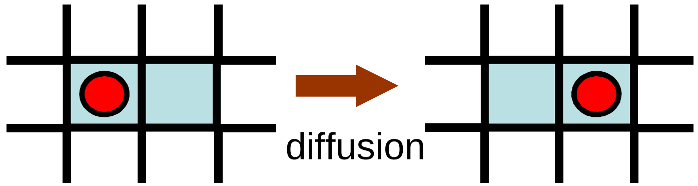


Illustration – DLA

- Diffusion Limited Aggregation: 2 evolution rules
  - ✓ A set of particles in some space
    2 types of particle: fixed (**F**) and mobile (**M**)
  - ✓ Local evolution laws
    - *Diffusion*: the particules **M** follow a random walk rule
    - *Aggregation*: if a part. **M** meets a part. **F**, it becomes **F**

- Expression in MGS






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

Neighborhood = interaction

Empty Place

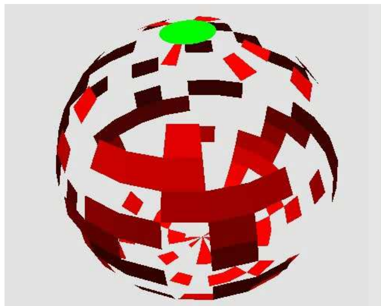
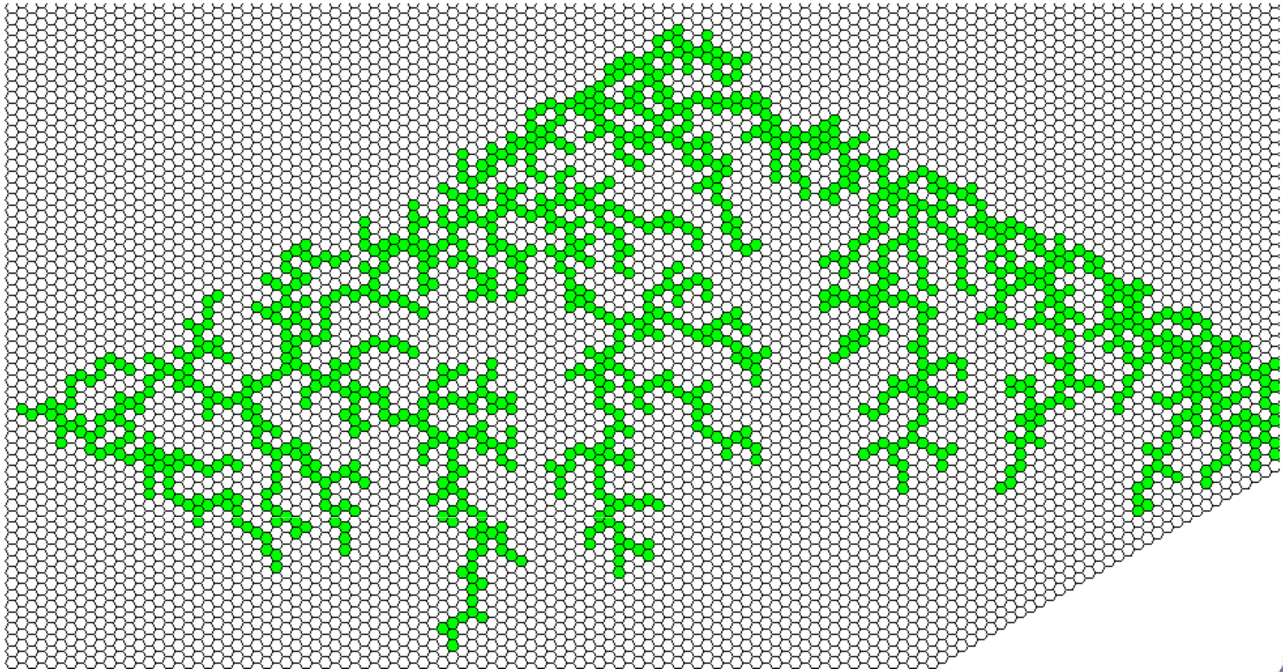


Illustration – DLA



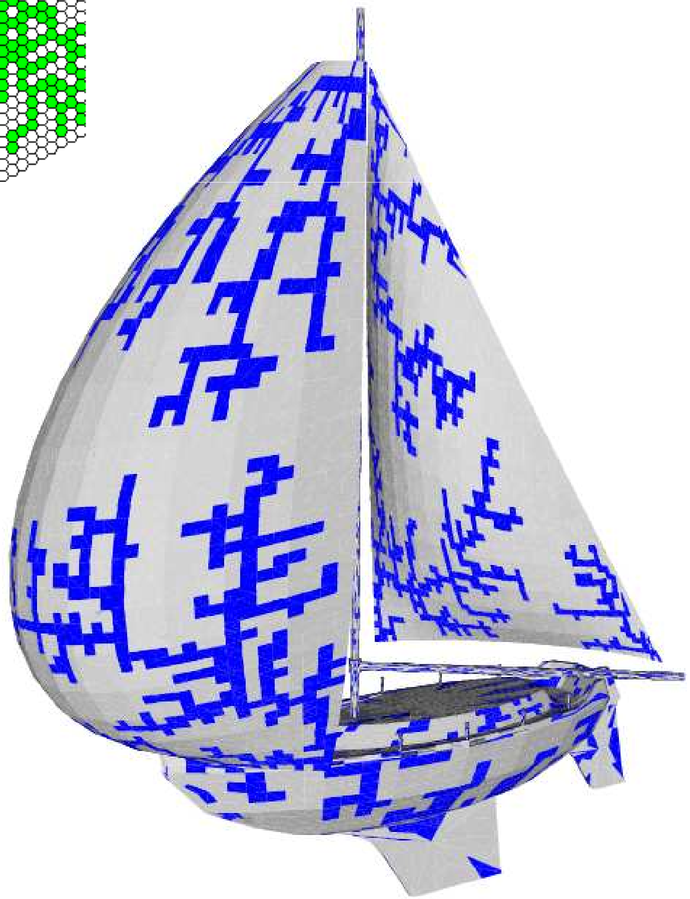
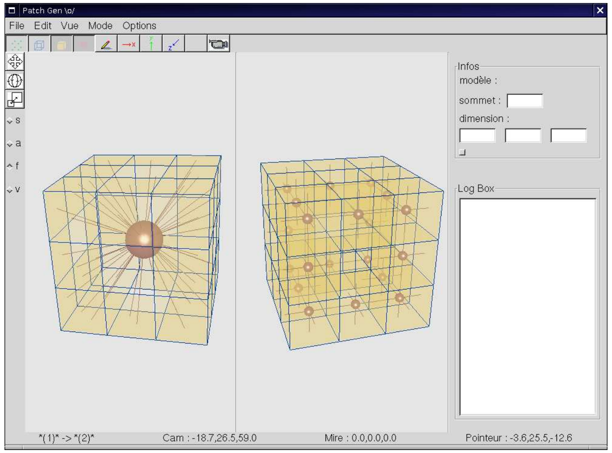


Illustration – Geometrical Modeling

Mesh refinement

- ✓ Cells subdivision
- ✓ Intuitively described locally
- ✓ Polyhedral Subdivision

  Neighborhood modification



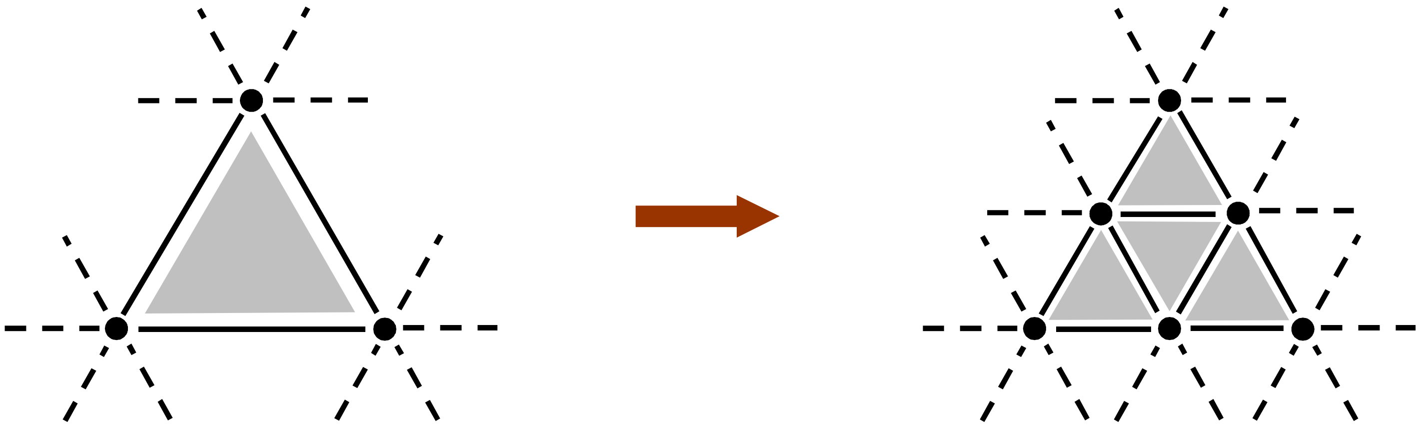


Illustration – Geometrical Modeling

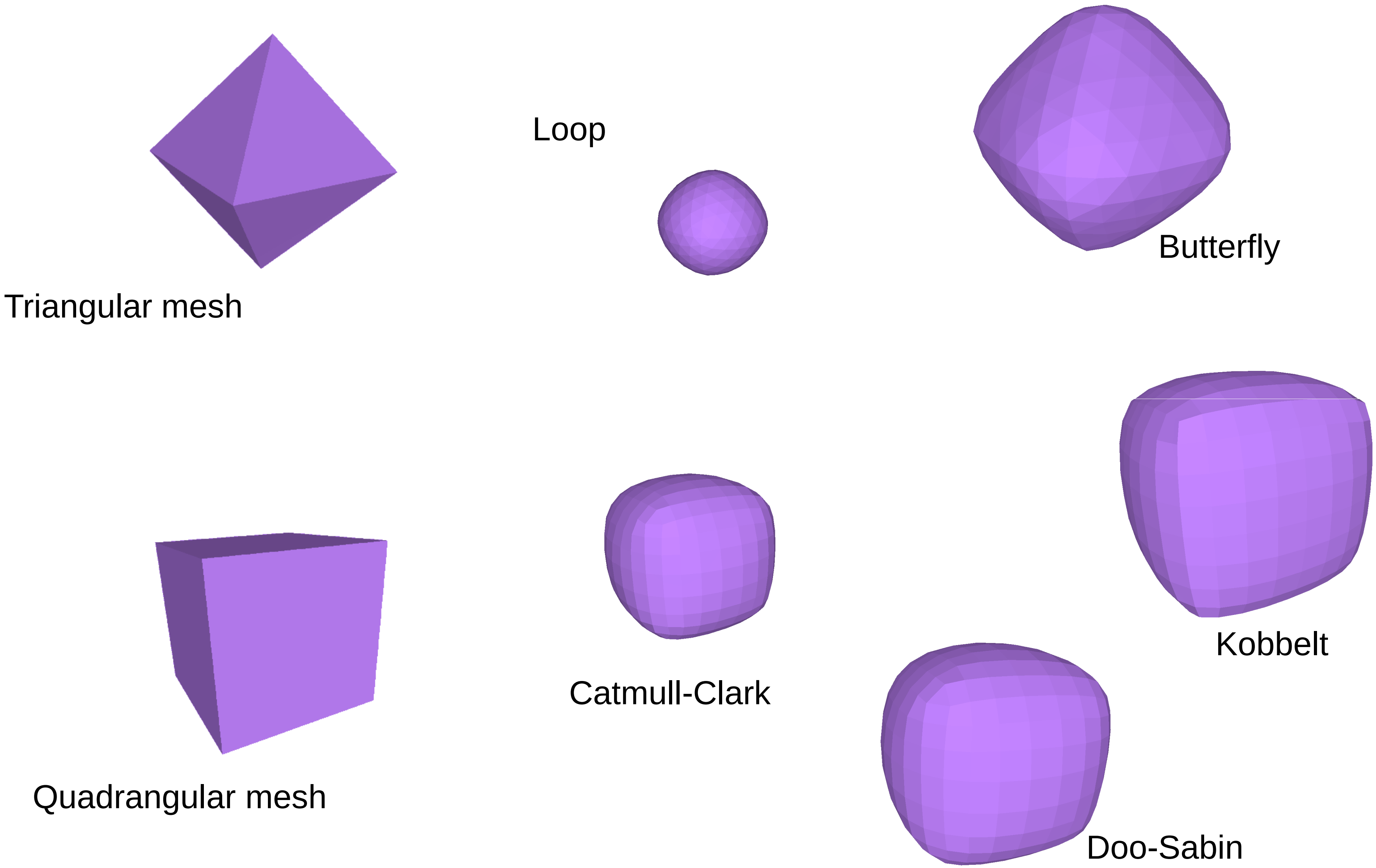
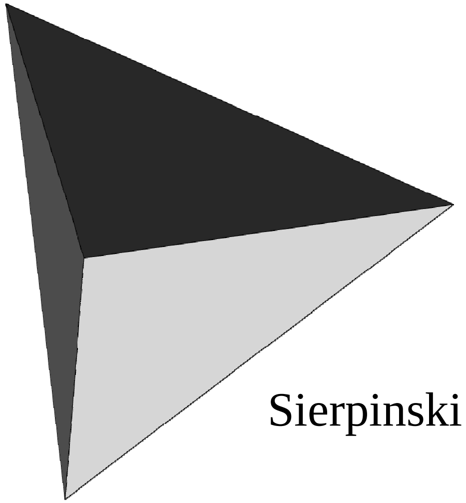


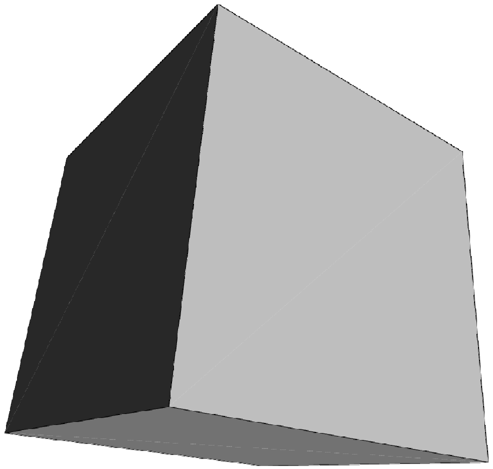
Illustration – Self-(dis)assembly

Carving



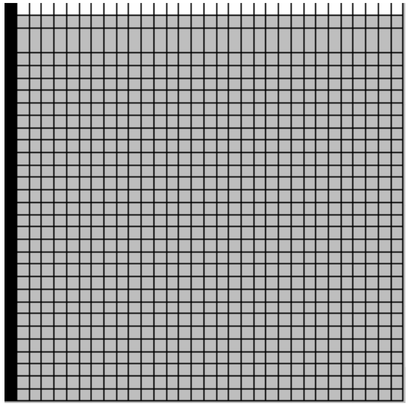
Sierpinski sponge

Menger sponge

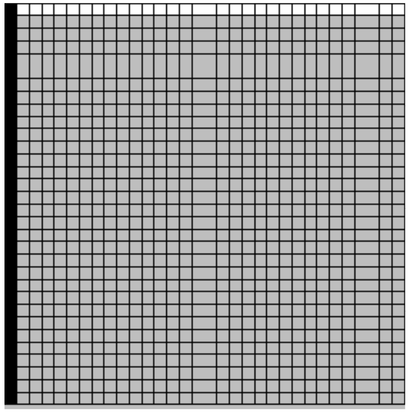


Accretive growth

(inspired by [Rothemund 2004] DNA self assembly)

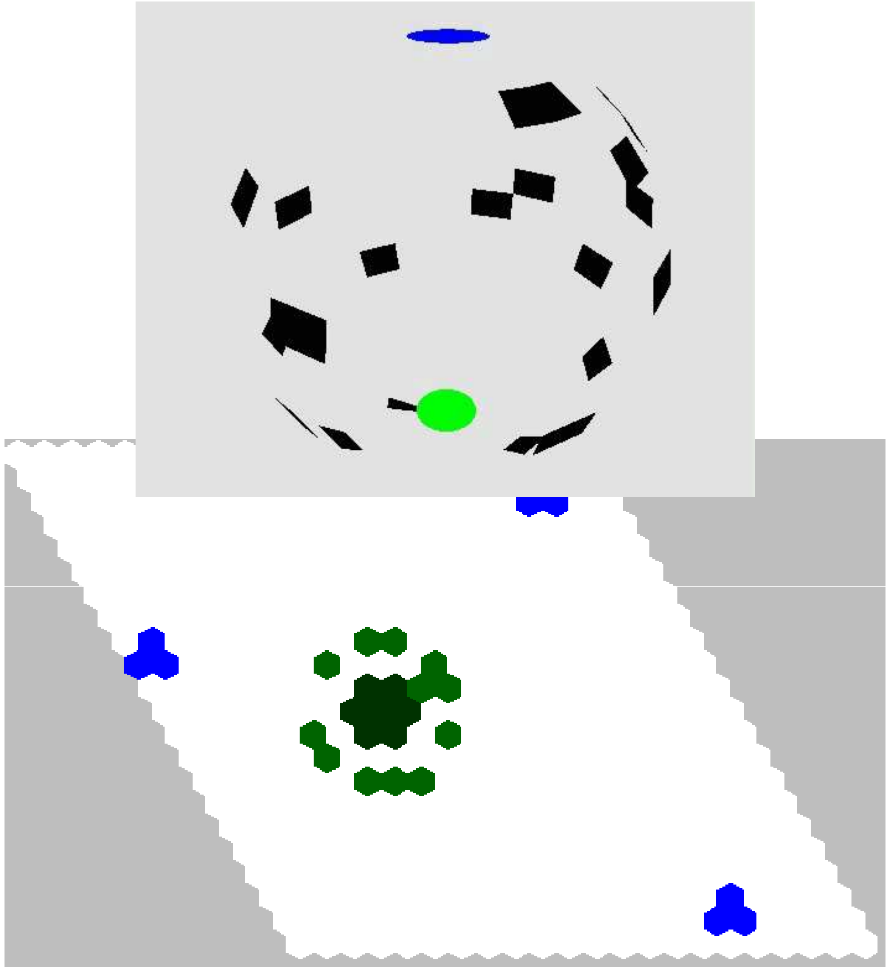


Maximal parallel strategy



Stochastic strategy

Programming a population

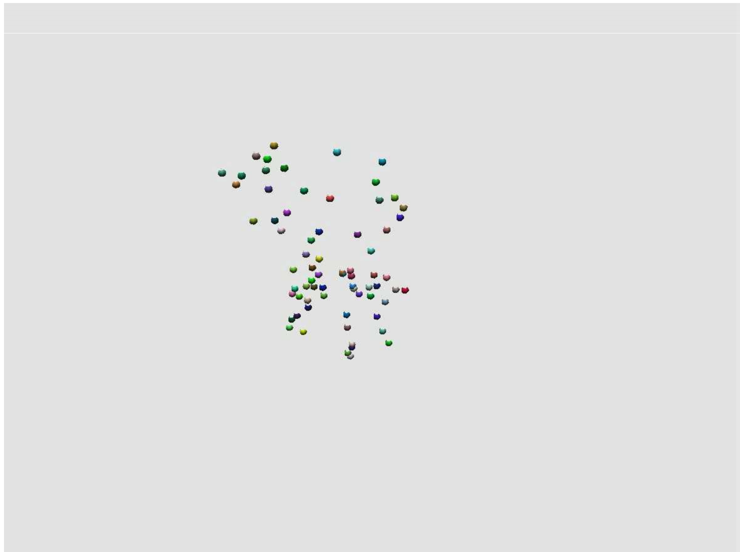


Ants foraging

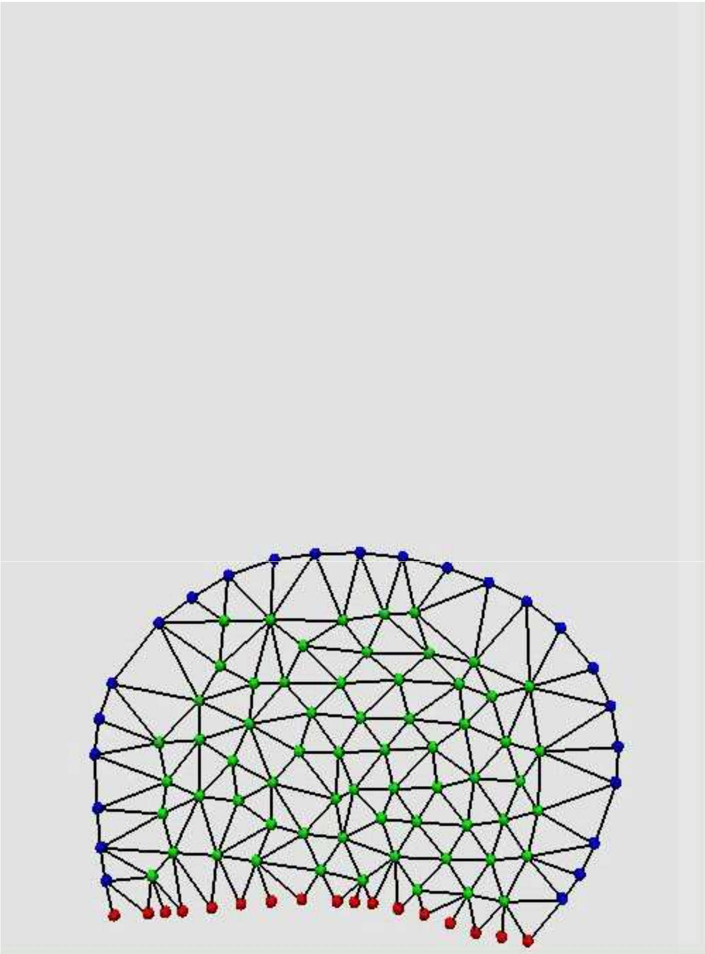
One transformation for any topologies: ***polytypism***

Flocking birds
(following Reynolds' Boids 87)

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



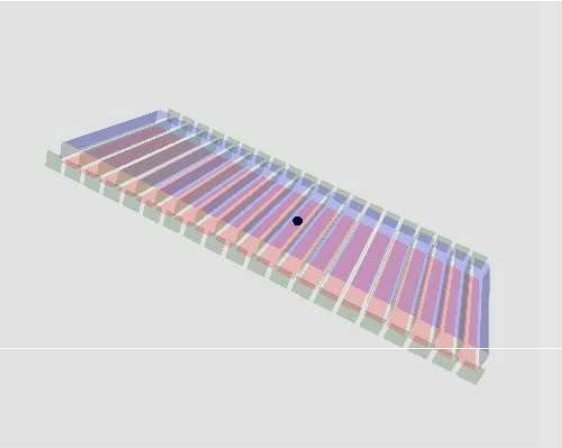
Dynamical Structures in Biology

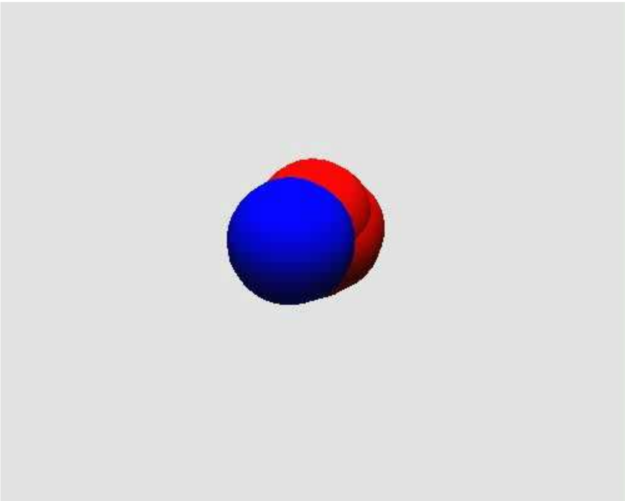


Cellular motility
Adaptive mesh

Neurulation
(following [Odell et al. 1981] and [Nagpal 2001])

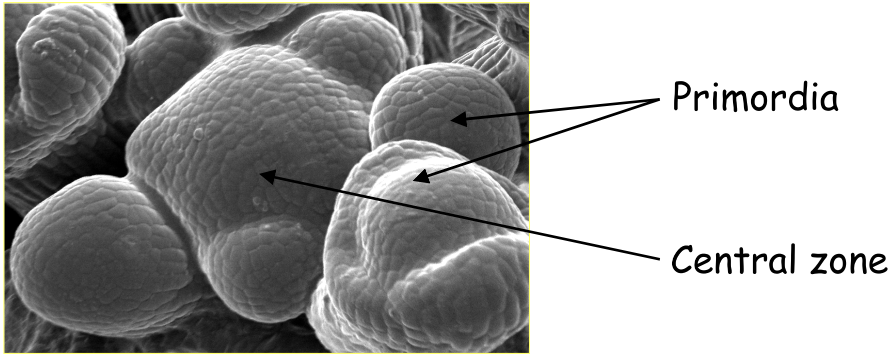
Modification of topology





Tumor growth

Modeling the Growth of a Meristem



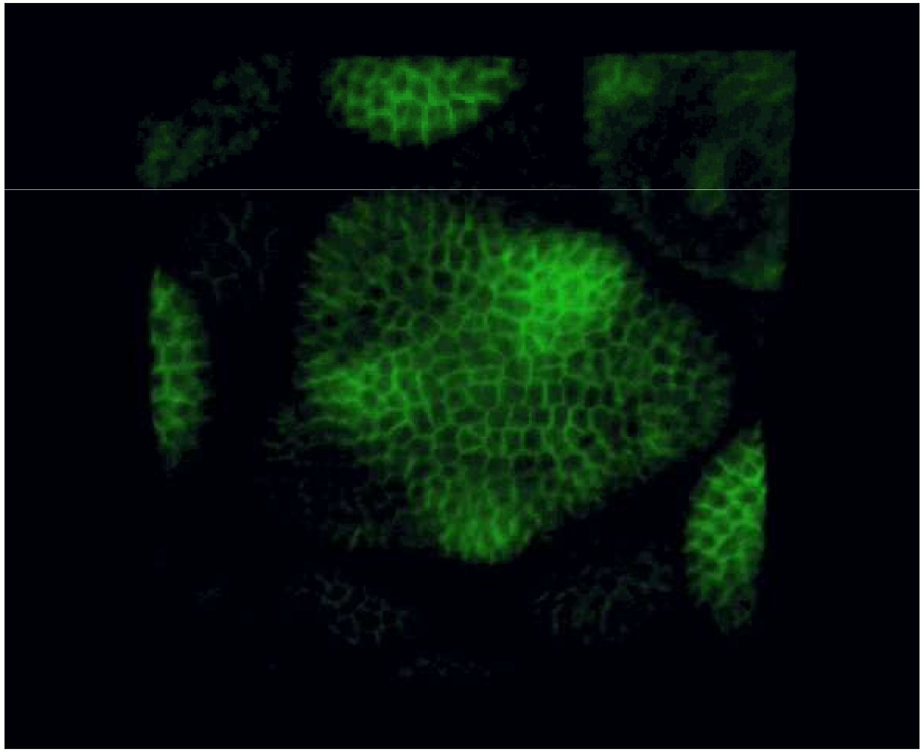
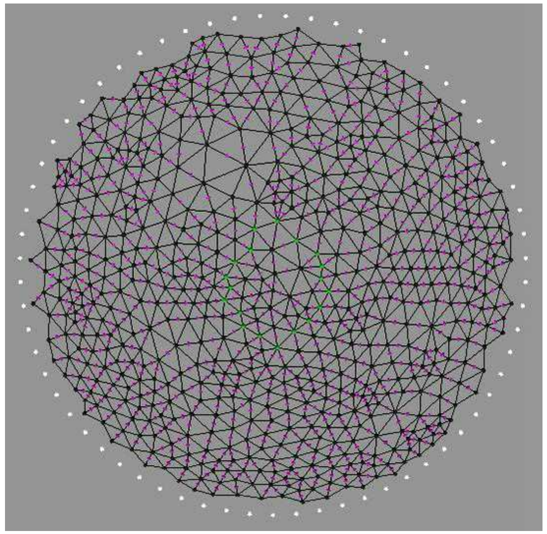
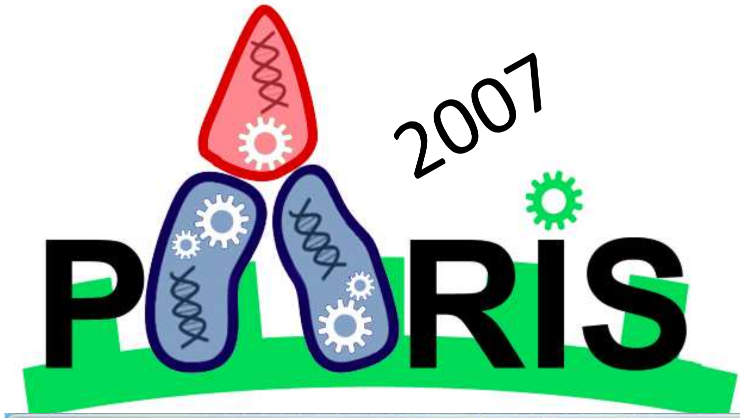


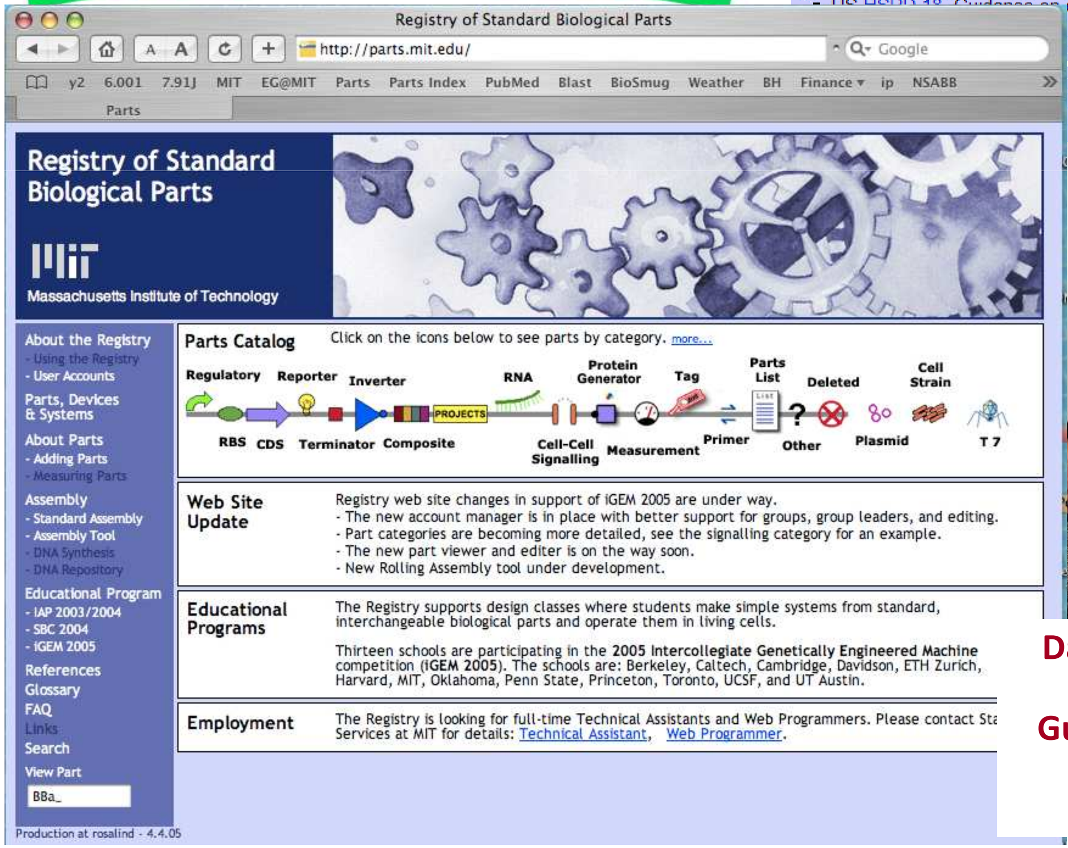
Image sequence showing cell division patterns via membrane-bound PIN1, in Shoot Apical Meristem (SAM), nearby floral meristems, and the boundaries between them.
http://computableplant.ics.uci.edu/



MGS Simulation
P. Barbier de Reuille
(with J. Traas & C. Godin) [PNAS 2007]

Synthetic Multicellular Bacterium





David Bikard, Thomas Landrain, David Puyraimond, Eimad Shotar, Gilles Vieira, Aurélien Rizk, David Guegan, Nicolas Chiaruttini, Thomas Clozel, Thomas Landrain

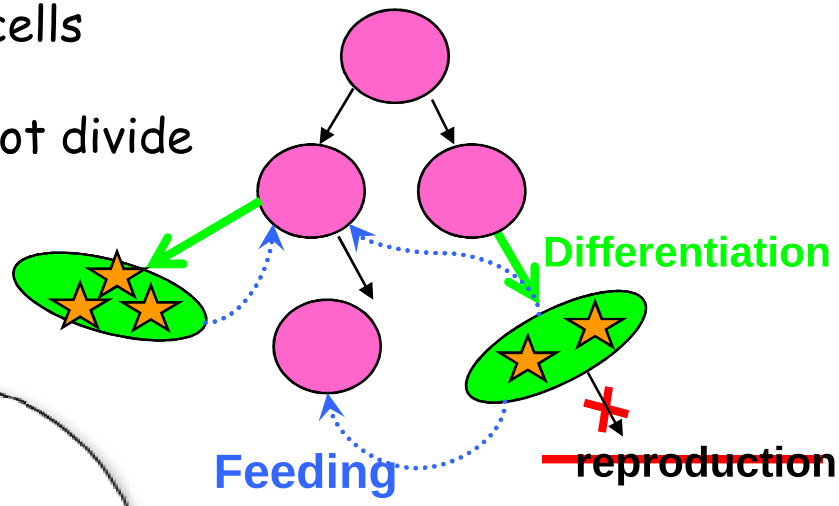
Synthetic Multicellular Bacterium

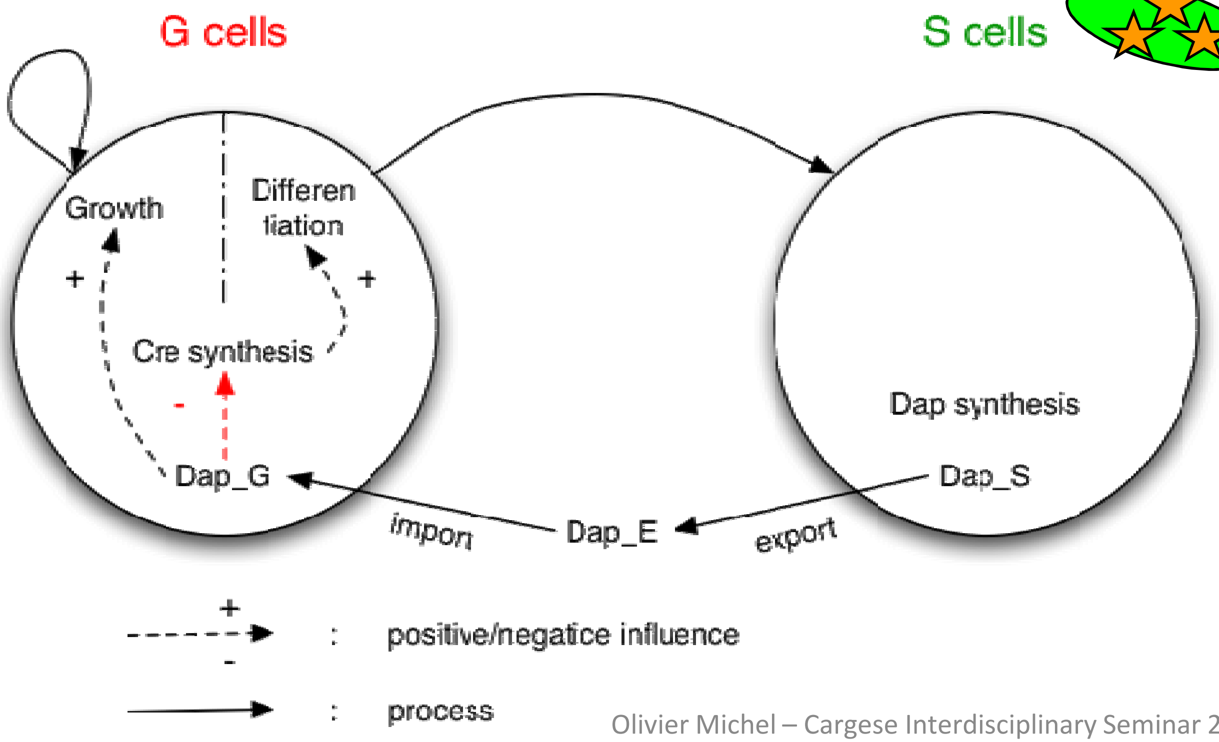
The Paris iGEM project:

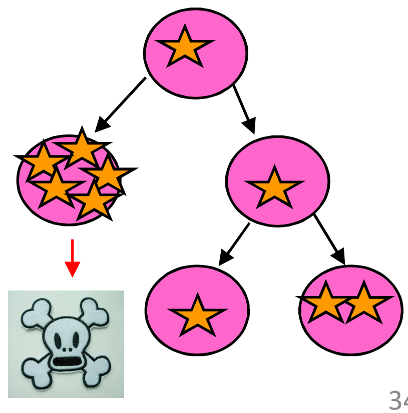
decoupling growth and transgene expression

germ cells can grow and divide only if metabolite (DAP) is present
differentiate into soma cells

soma cells produce a metabolite (DAP), cannot divide

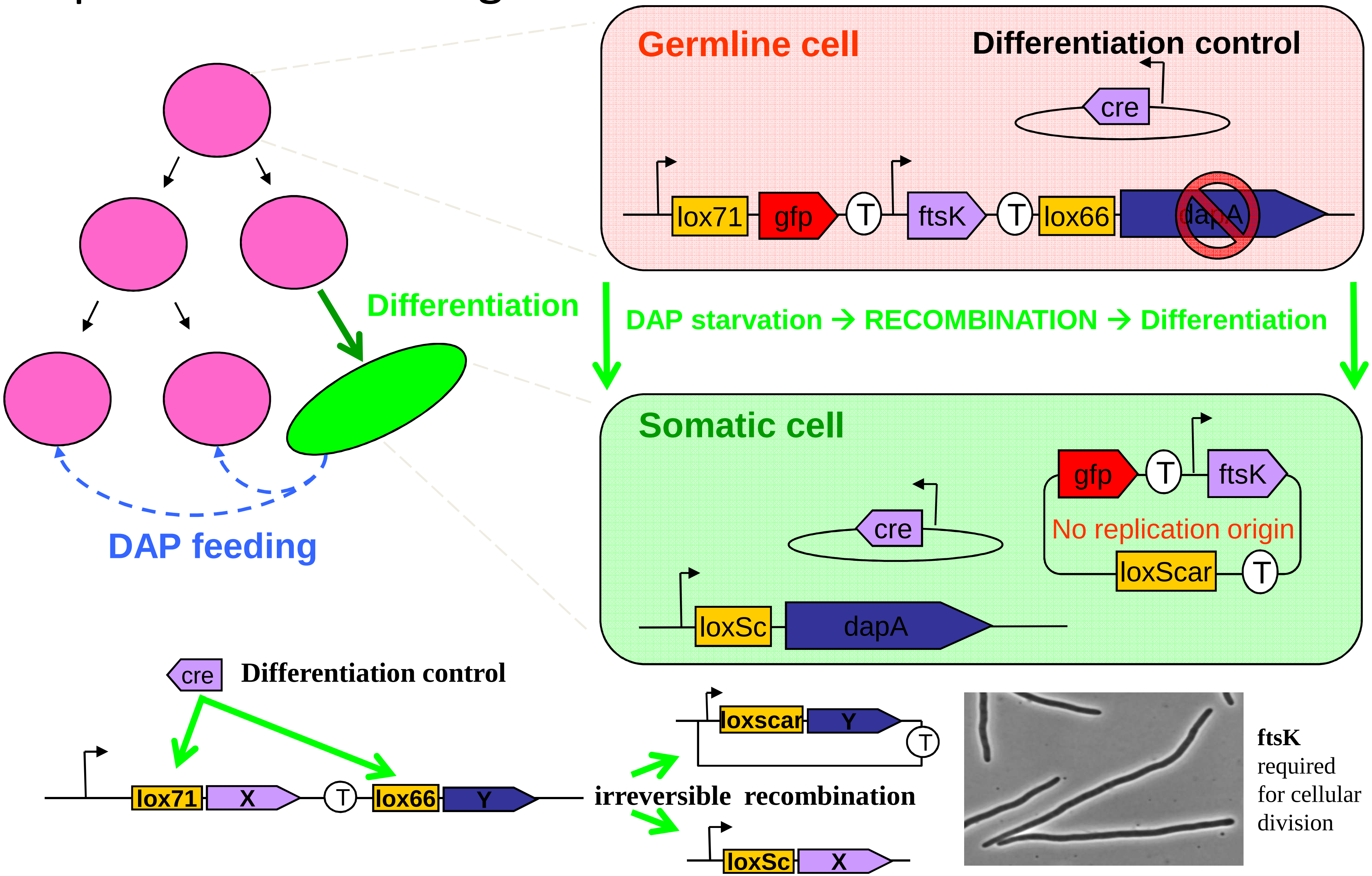






Synthetic Multicellular Bacterium

Implementation using Bio-Bricks

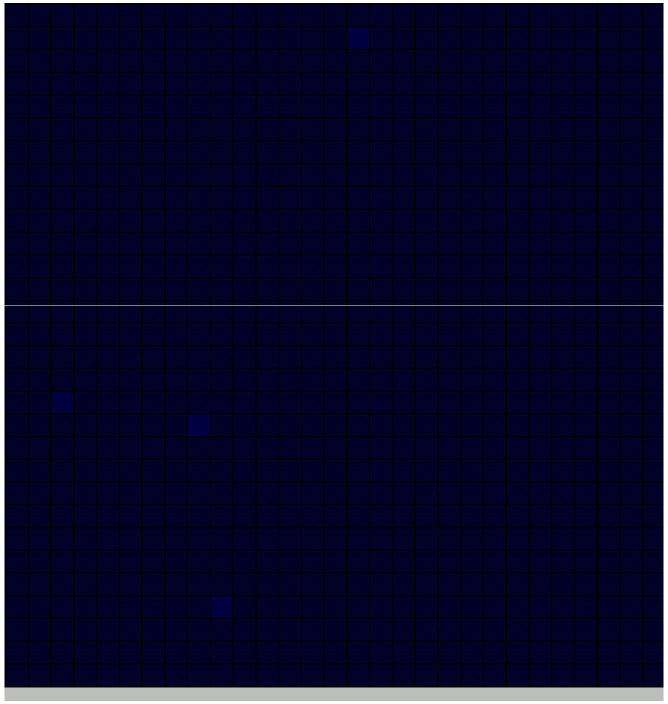


ftsK required for cellular division

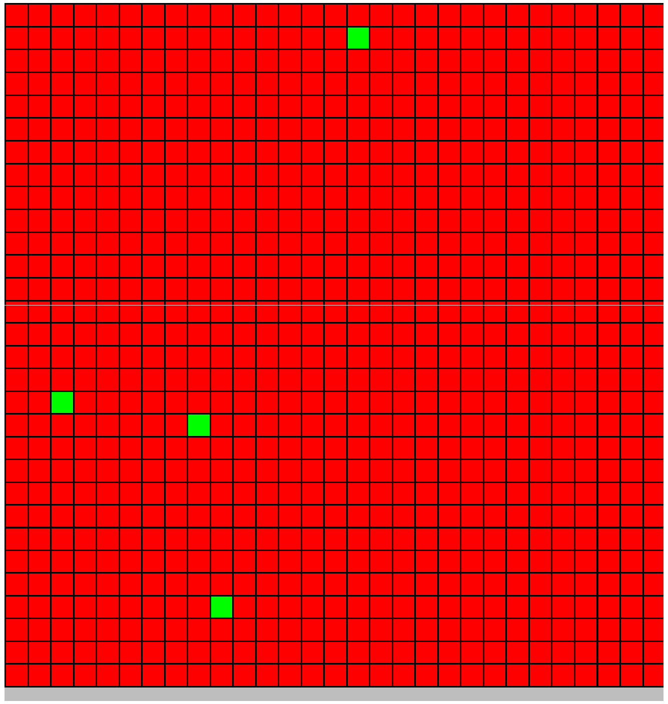
Synthetic Multicellular Bacterium

- Simulation to answer 4 questions

How does differentiation induces feeding? (proof of concept)
(phenomenological) 30x30 2D toric cellular automaton (in MGS)



diffusion of DAP by somatic cells



Differentiation of somatic and ge

⇒ crucial importance of 3 communications processes:

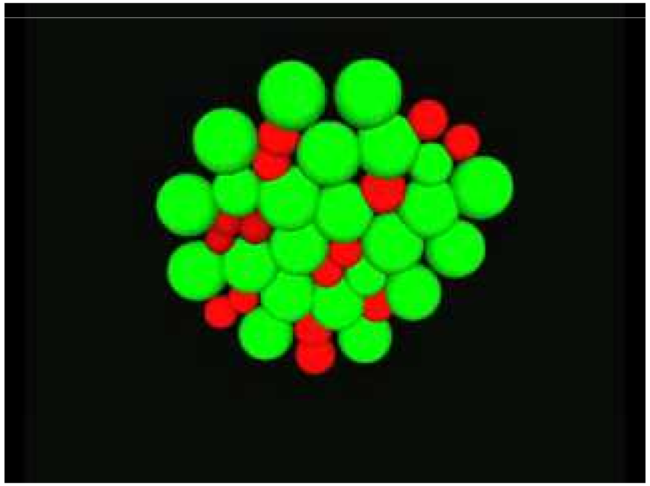
- export
- diffusion (high diffusion speed is essential)
- import (high production rate is required while importing)

Synthetic Multicellular Bacterium

- Simulation to answer 4 questions

How does differentiation induces feeding? (proof of concept)
(phenomenological) 30x30 2D toric cellular automaton (in MGS)
How do spatial organization and distribution evolve?
(phenomenological) agents based mass-spring system on a Delaunay (in MGS)

Such model allows:
- *division* and *death* by adding or removing masses
- cell *growth* by increasing springs rest length,
- *holes filling* in the population
  (in empty places in the population, springs push masses to fill it)
- prevents *cells dispersion*
  (springs cannot infinitely extends)



somatic and germ cell
germ cells *grow*, *divide* and *differentiate*
somatic cells produce DAP that *diffuses*

Synthetic Multicellular Bacterium

- Simulation to answer 4 questions

How does differentiation induces feeding? (proof of concept)
(phenomenological) 30x30 2D toric cellular automaton (in MGS)
How do spatial organization and distribution evolve?
(phenomenological) agents based mass-spring system on a Delaunay (in MGS)
How robust and tunable is the model?
(phenomenological) ODE kinetics (by hand and Matlab)

S
- c
- t

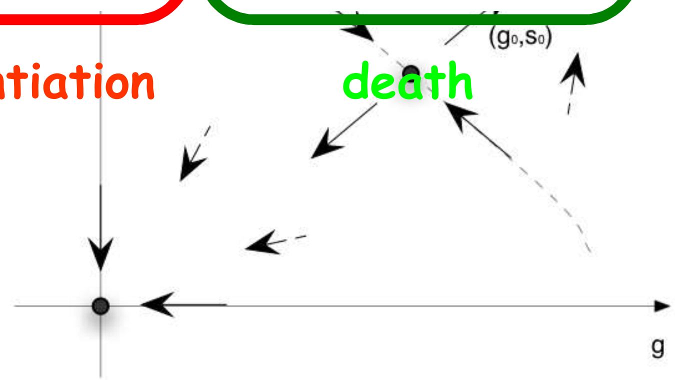
$$\frac{d[G]}{dt} = \lambda_{div}\frac{[S]}{\theta_{div}+[S]}[G] - \lambda_{diff}[G] - \lambda_{death}[G]$$

$$\frac{d[S]}{dt} = \lambda_{diff}[G] - \lambda_{death}[S]$$

growth — differentiation — death

$\Rightarrow$ population size remains constant

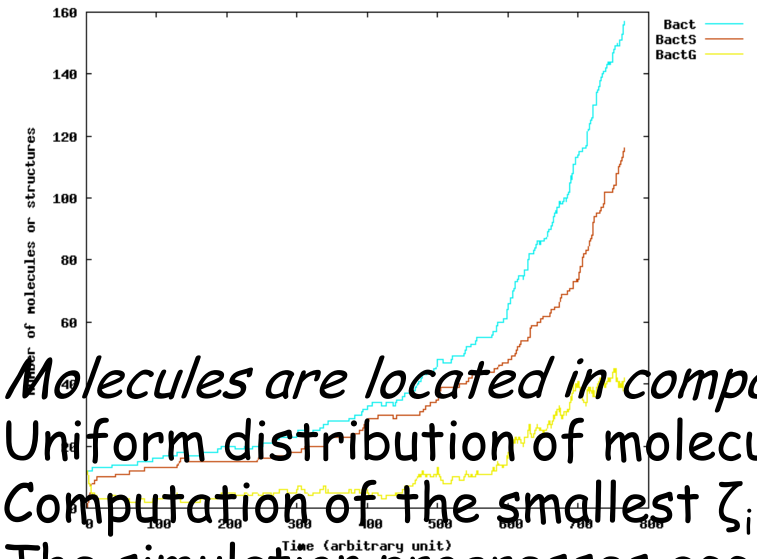
- for higher values of the initial conditions
  $\Rightarrow$ we expect exponential growth of the populations



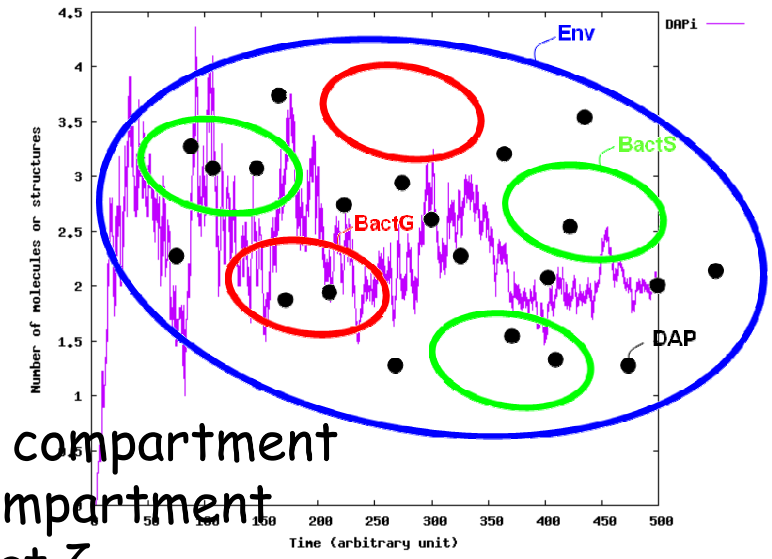
Synthetic Multicellular Bacterium

- Simulation to answer 4 questions

How does differentiation induces feeding? (proof of concept)
(phenomenological) 30x30 2D toric cellular automaton (in MGS)
How do spatial organization and distribution evolve?
(phenomenological) agents based mass-spring system on a Delaunay (in MGS)
How robust and tunable is the model?
(phenomenological) ODE kinetics (by hand and Matlab)
How sensitive is the system to noise?
(phenomenological) Gillespie's SSA based simulation (in MGS)



exponential growth of bacteria population



mean number of DAP molecules in a cell (<5)

Molecules are located in compartments
Uniform distribution of molecules in each compartment
Computation of the smallest ζ_i for each compartment
The simulation progresses according to that ζ_i

Conclusion and Future Works

- **MGS**, a language for the modeling and simulation of DS^2
  - ✓ Topological Collections
    - Data field on a topological space
  - ✓ Transformations
    - Topological rewriting on topological collections – interactions – strategies for handling of time
  - ✓ Generalization of various computing models (chemical computing, P systems, L-systems, DNA computing, cellular automaton)
- Multi-level/multi-scale/hybrid modeling
  - ✓ Diffusion process described in continuous/discrete terms
- SYNthetic BIOlogical sysTems: from desIgn to Compilation (SYNBIOTIC)
  - ✓ Using MIT’s bio-brick ontology
  - ✓ Iterative compiling high-level programs to bio-ware
  - ✓ Shape engineering for a population of cells
  - ✓ Spatial/Amorphous/Autonomic - computing

Long-term Goal: Synthetic Biology

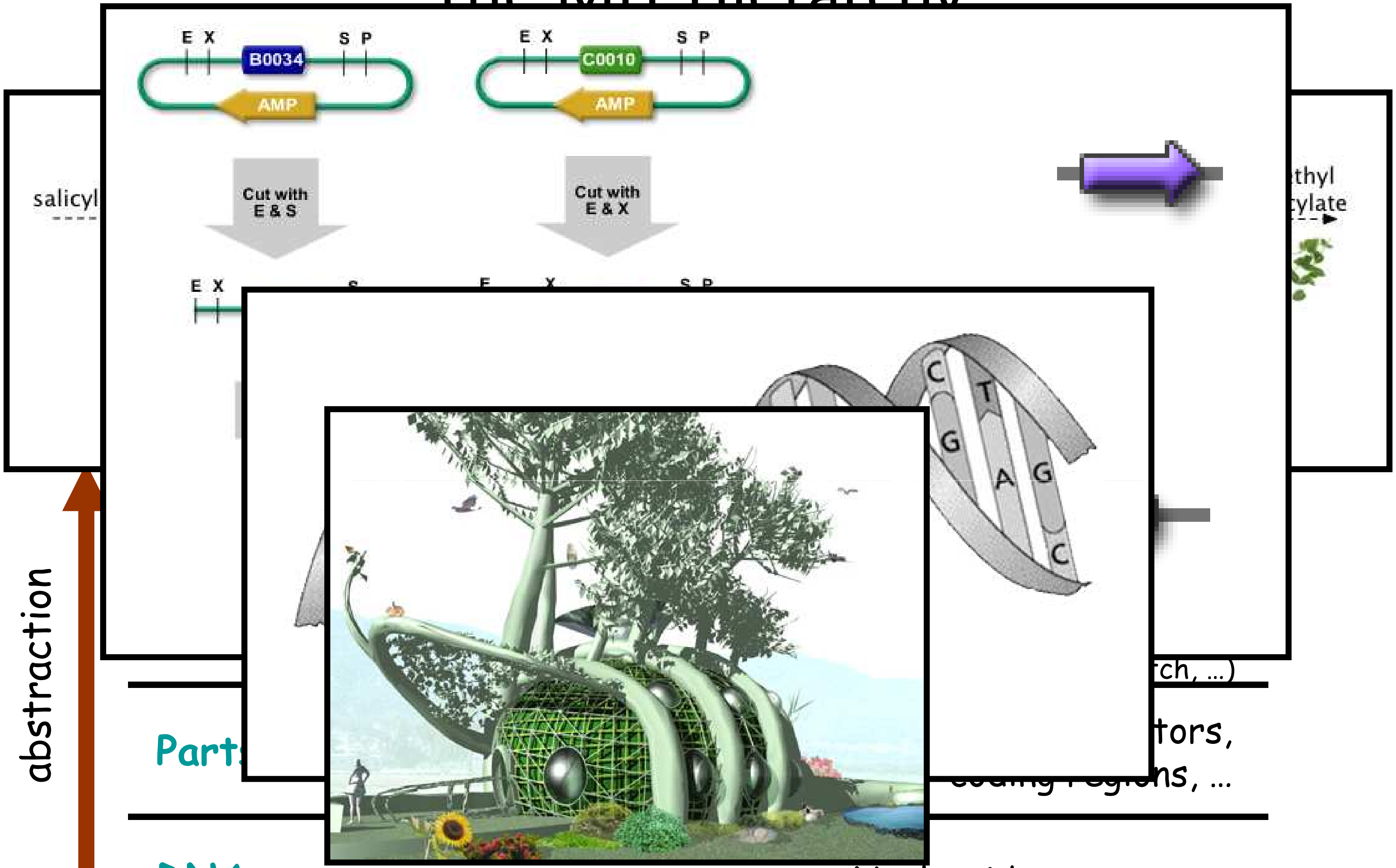
➡ Programming biological processes

When biology meets engineering principles…

- ✓ standardization (definition of functions, standards, libraries, …)
- ✓ abstraction (organization of functions by levels)
- ✓ decoupling (conception/realization)

…for the design of new biological functions by *easy* and *standard* composition of building-blocks

The MIT Hierarchy



abstraction

Parts

DNA

Nucleotides sequences

Specify Globally, Compute Locally

Spatial computing:
computing with fields, wave, gradients, *etc.*

Amorphous computing:
programming with asynchronous, unreliable agent

Synthetic biology:
cell and cell population

$$\frac{\partial \phi}{\partial t} = D\nabla^2 \phi(\vec{r}, t)$$