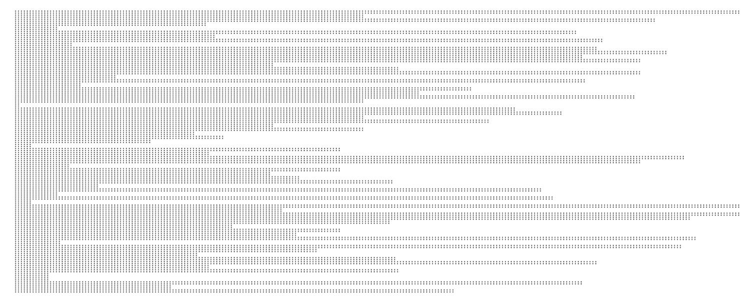
MGS: a declarative spatial computing programming language

Jean-Louis Giavitto[a]
Antoine Spicher[b]
[a] **IRCAM – CNRS**
[b] **LACL – Université de Paris Est**

http://mgs.spatial-computing.org

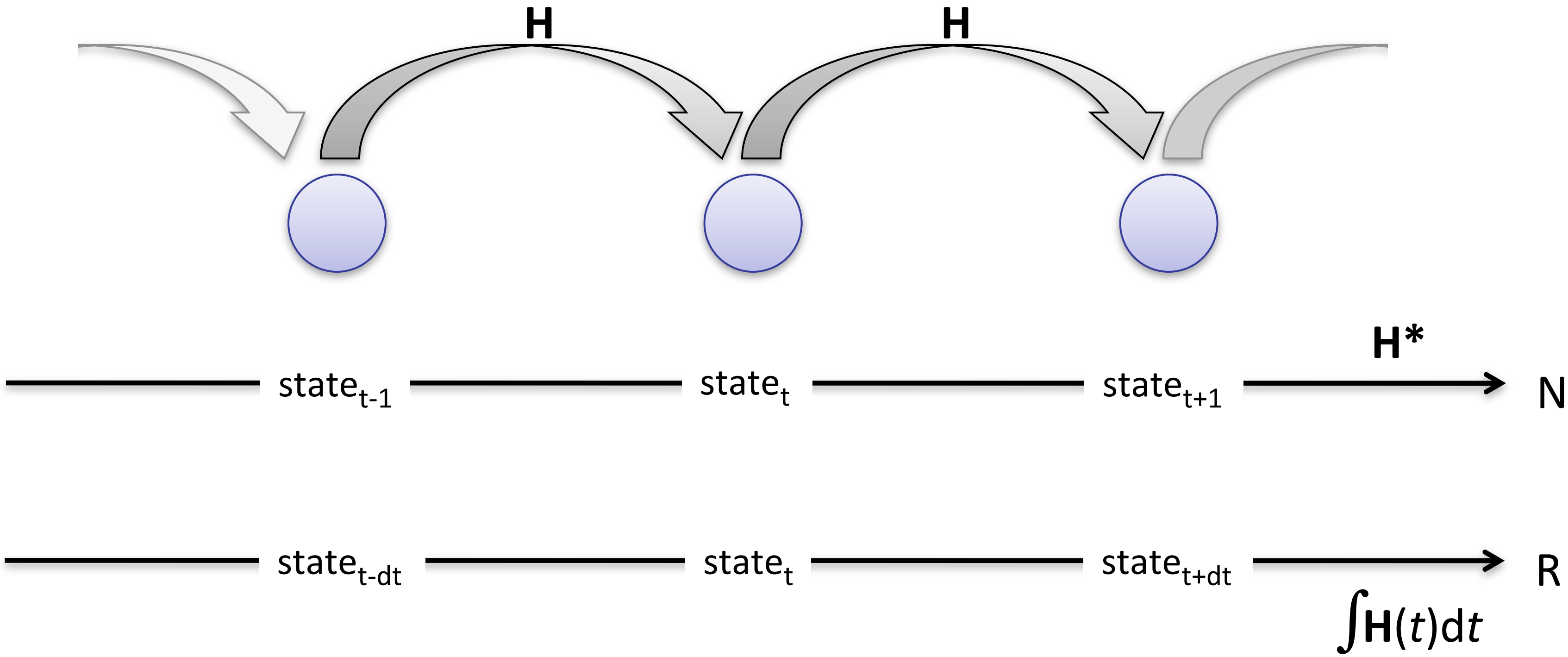


We sketch the rationals of the MGS programming language. MGS is an experimental programming language developed to study the application of several spatial computing concepts to the specification and simulation of dynamical systems with a dynamical structure. MGS extends the notion of rewriting by considering more general structure than terms. The basic computation step in MGS replaces in a topological collection A, some subcollection B, by another topological collection C. A topological collection is a set of element structured by a neighborhood relationships describing an underlying space representing a data structure or constraints that must be fulfilled by the computation. This process proposes a unified view on several computational mechanisms initially inspired by biological or chemical processes (Gamma and the CHAM, Lindenmayer systems, Paun systems and cellular automata).

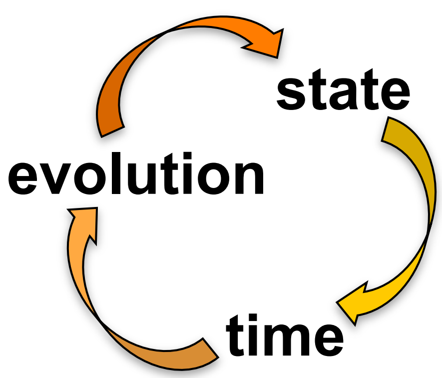
1. $(DS)^2$
2. Gamma, P systems, L systems, cellular automata…
3. Spatial generalization
4. MGS
5. Algorithmic examples
6. Biological modeling

Dynamical systems
and
Dynamical Structures

Specifying a dynamical system (for simulation)



Specification of
- **structure of state**
- **structure of time**
- **evolution function**



Formalism for Dynamical System

MGS

- State : often structured by space (e.g. fields)
- Time
- Evolution function

C : continuous, D: discrete	PDE	Coupled ODE	Iteration of functions	Cellular automata	…
state	C	C	C	D	…
time	C	C	D	D	…
space	C	D	D	D	…

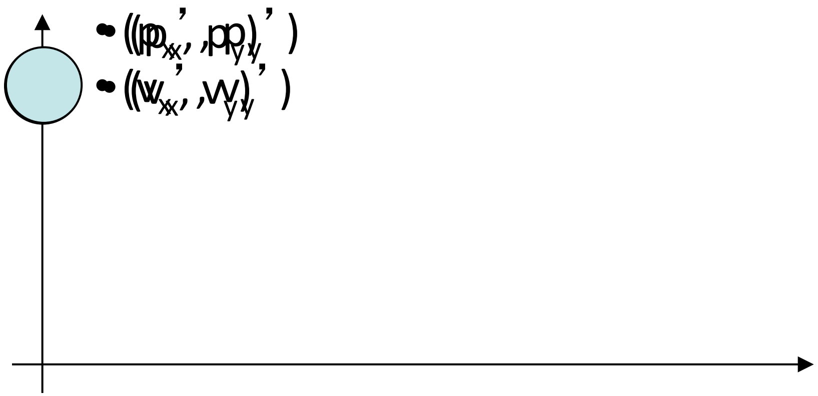
The medium/process problem

THE CHEMICAL BASIS OF MORPHOGENESIS

BY A. M. TURING, F.R.S. *University of Manchester*

(*Received* 9 *November* 1951—*Revised* 15 *March* 1952)

a falling ball



at any time a state is a position and a speed

A dynamical system (DS)

The medium/process problem

THE CHEMICAL BASIS OF MORPHOGENESIS

BY A. M. TURING, F.R.S. *University of Manchester*

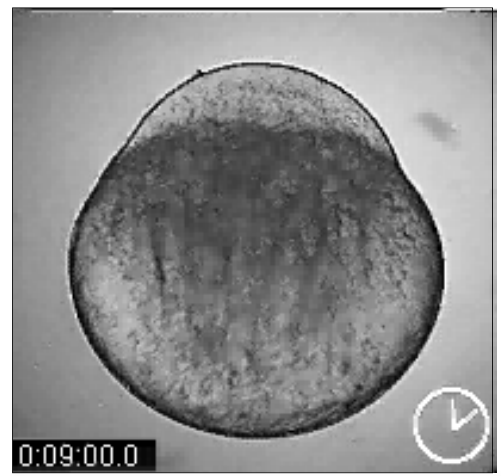
(*Received* 9 *November* 1951—*Revised* 15 *March* 1952)

a falling ball

at any time a state is a position and a speed

A dynamical system (DS)

a developing embryo

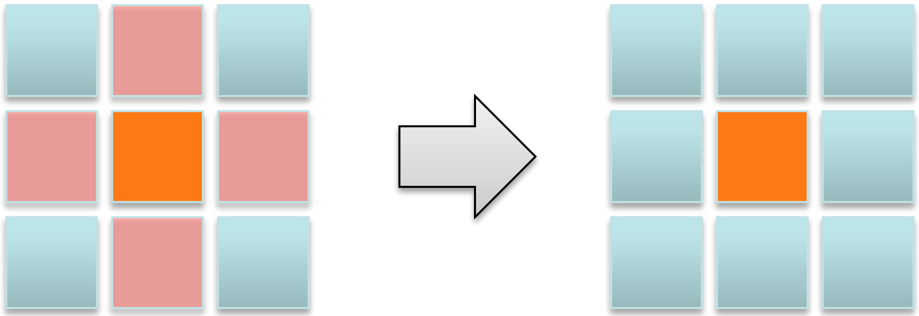


the structure of the state is changing in time (chemical and mechanical state of each cell)

A dynamical system with a dynamical structure $(DS)^2$

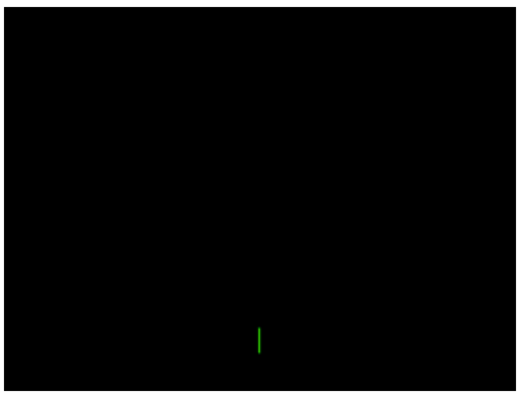
Bio-inspired models of $(DS)^2$

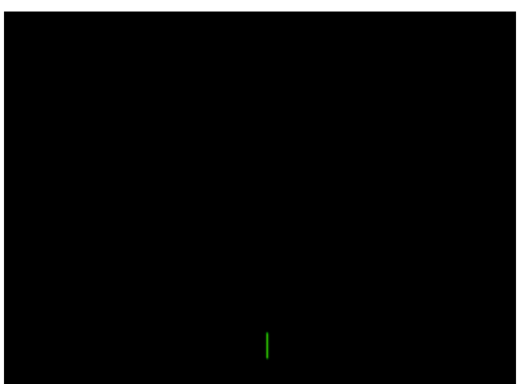
- Von Neumann
- Avoids the dynamic structure problems
  - Predefined underlying (unbounded) space
- Replace a cell *X*
  in an NEWS grid
  by another one (with a new state)



Lindenmayer systems

It might be possible, however, to treat a few particular cases in detail with the aid of a digital computer. This method has the advantage that it is not so necessary to make simplifying assumptions as it is when doing a more theoretical type of analysis. It might even be possible to take the mechanical aspects of the problem into account as well as the chemical, when applying this type of method. The essential disadvantage of the method is that one only gets results for particular cases. But this disadvantage is probably of comparatively little importance.





P. Prusinkiewicz

Lindenmayer systems

- The structure of a tree can be coded by a string of parenthetised symbols
- A symbol is an elementary part of the plant
- The symbol between [and] represents a sub-tree
- Additional conventions are used to represent main axis, orientation, depth, etc.
- A rule

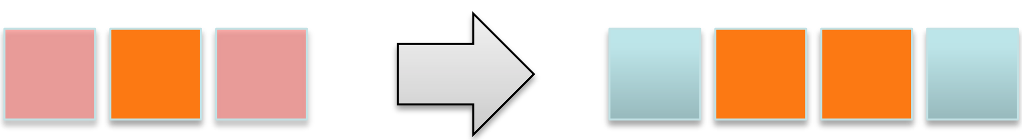
  $$s_0 \rightarrow s_1\ s_2\ s_3 \ldots$$

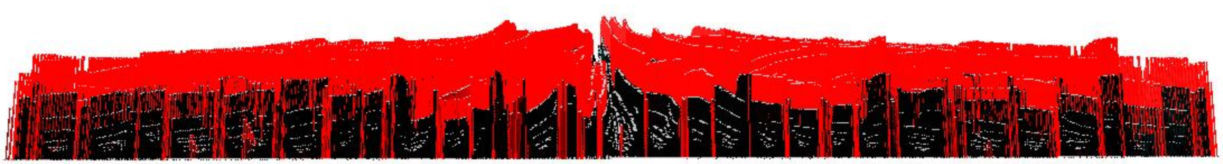
  represents the evolution of s_0

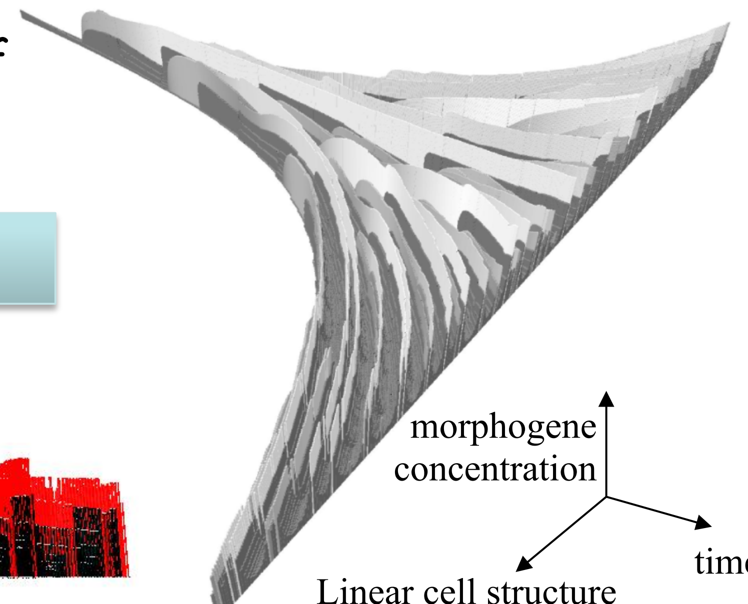
Lindemayer Systems

- Replace a substring *X*
  in a string
  by another one

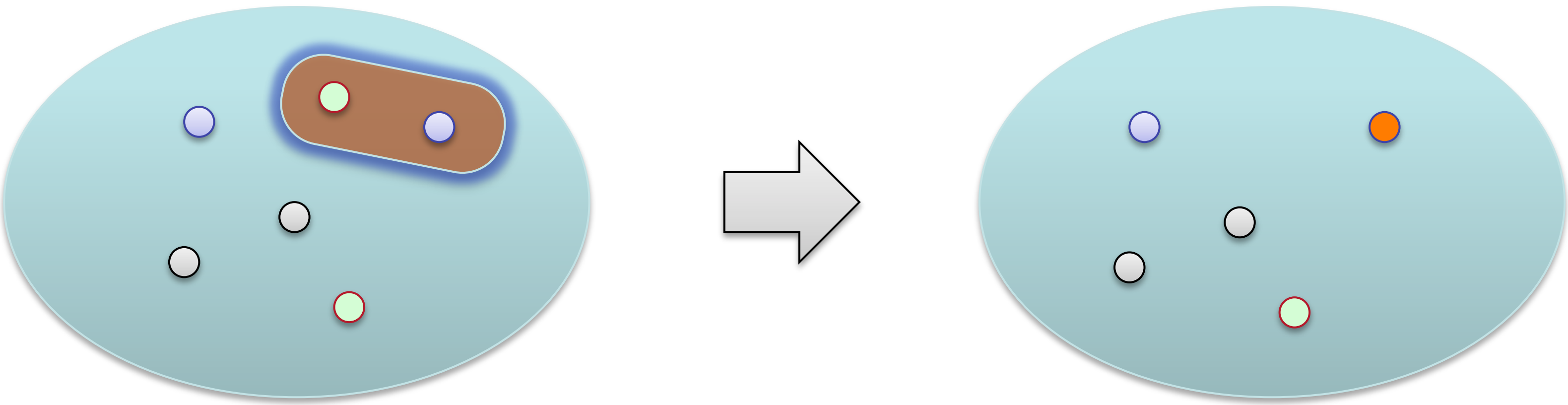
```
e / (C(e) & (e.x >= lm) & (e.p == L))
=> {type="C", a=e.a, h=e.h, x=e.x*shorter, p=L},
   {type="C", a=e.a, h=e.h, x=e.x*longer
```

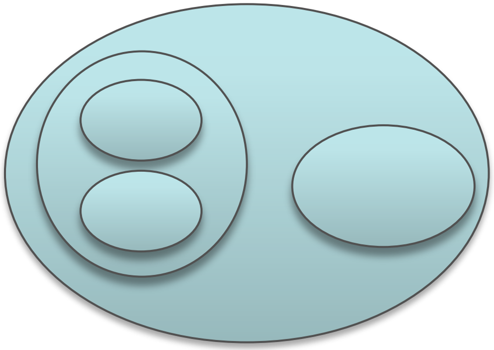






- Replace a sub-multiset *X*
  in a multiset
  by another one





A general device

A general device?

A general rewriting mechanism

1. In a *collection* of elements
2. *Replace* a *subcollection* X
3. With a collection Y computed from X and its *neighbors*

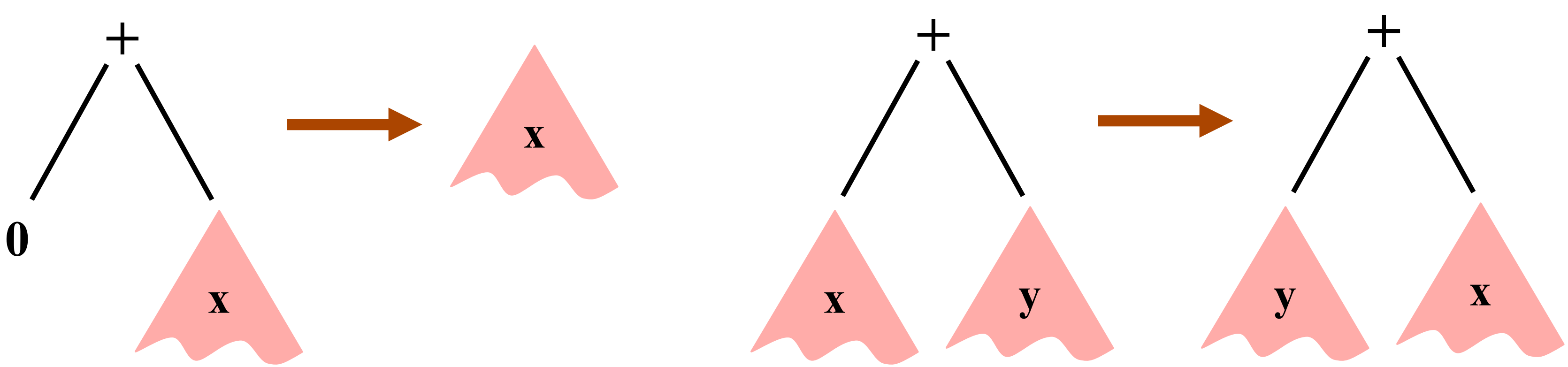
Collection	**Neighborhood**	**data structure**
Term	father/son	free
Set	all	A + C + Idempotent
Multiset	all	A + Commutative
Sequence	left and right (with jump for trees)	Associative
Commutative		
Grid	NEWS	

monoidal

Rewriting systems (and abstract transition systems)

- Rewriting system
  - Used to formalize equationnal reasoning
  - A generative device (grammar)
  - Replace a sub-part of an entity by an other
  - Set of rewriting rules $\alpha \rightarrow \beta$
    - α: pattern specifying a sub-part
    - β: expression evaluating a new sub-part

- Example: arithmetic expressions simplification



Rewriting on terms+property

$1 + 2 \rightarrow \ldots$ (arithmetic) term rewriting

arithmetic operation

$a \,.\, b \rightarrow \ldots$ string rewriting (~ L systems)

string concatenation: « . » is a formal associative operation

$2H + O \rightarrow H_2O$ multiset rewriting (~ chemistry)

multiset concatenation (= the chemical soup): « . » is AC

A general device?

A general rewriting mechanism

1. In a *collection* of elements
2. *Replace* a *subcollection* X
3. With a collection Y computed from X and its *neighbors*

- **Collection** **Neighborhood** **data structure**

	Collection	Neighborhood	data structure
	– Term	father/son	free
monoidal	– Set	all	A + C + Idempotent
monoidal	– Multiset	all	A + Commutative
monoidal	– Sequence Commutative	left and right (with jump for trees)	Associative
	– Grid	NEWS	

The Spatial Approach

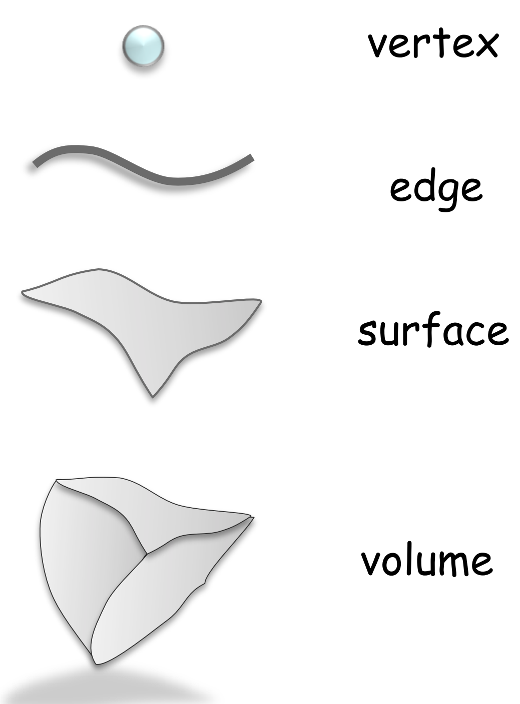
- Use space (topology) to unify the various collection structures
  - space as as a resource
  - space as a constraint
  - space as an input/output
- Neighborhood relationships:
  - the structure of the collection
  - the structure of the subcollection
  - the computation dependencies
- Substitution (replacement) topological surgery

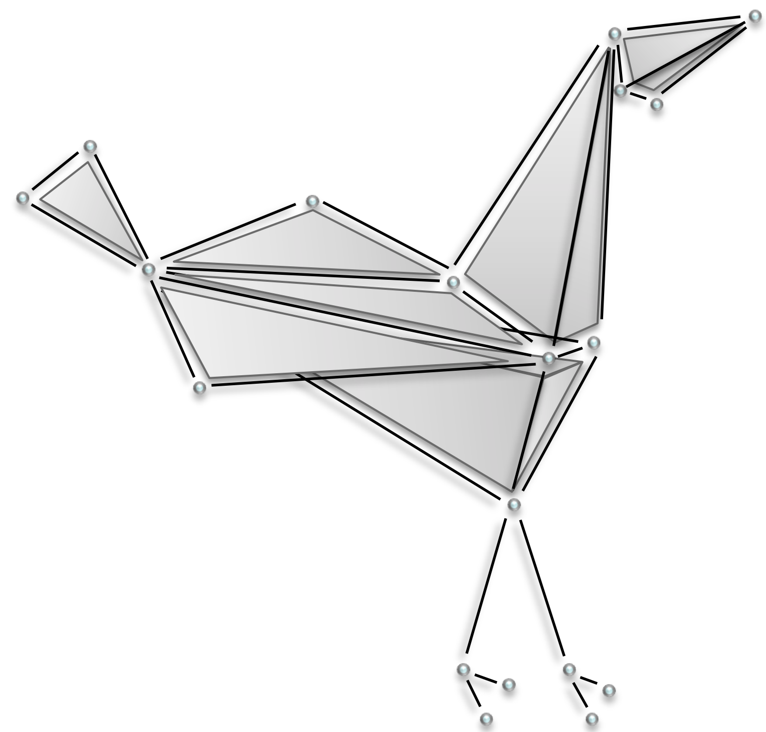
Properties

- *local evolution rules*
  mandatory when you cannot express a global function/relation because the domain of the function/relation is changing in time
- interaction based approach
  the l.h.s. of a rule specifies a set of elements in *interaction*, the r.h.s. the result of the interaction
- *the phase space is well defined but not well known*
  a generative process enumerates the elements but membership-test can be very hard
- *various kind of time evolution* (for the same set of rules)
- *demonstration by induction*
  on the rules or on the derivation (e.g. growth function in L system)

MGS Proposition

- Topological collections
  - Structure
    - A collection of topological cells
    - An ***incidence relationship***



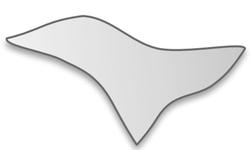


MGS Proposition

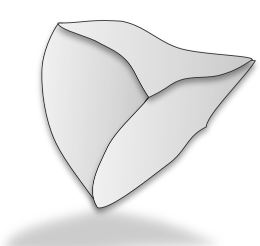
- Topological collections
  - Structure
    - A collection of topological cells
    - An incidence relationship
  - Data: **association of a value with each cell**

0-cell

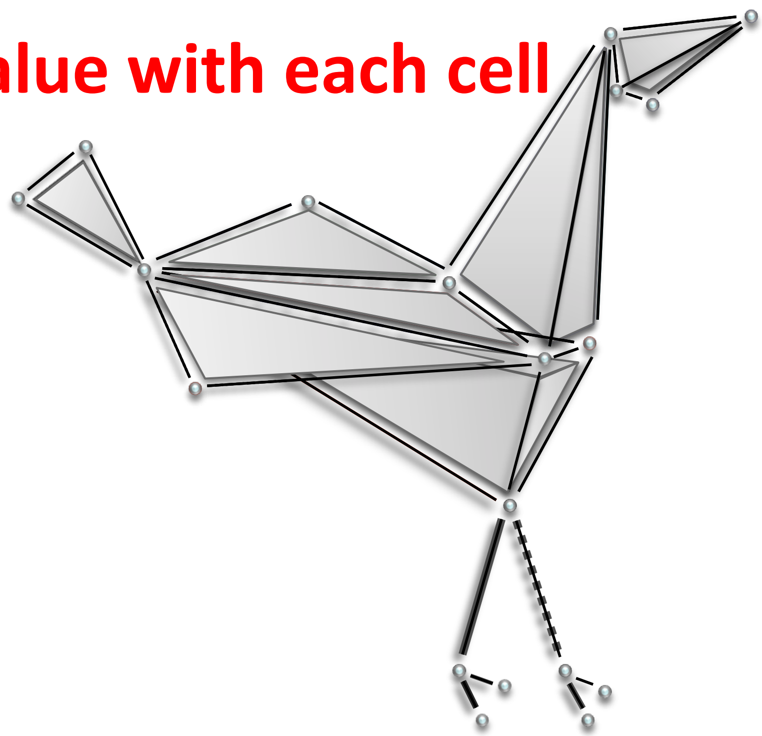
1-cell



2-cell



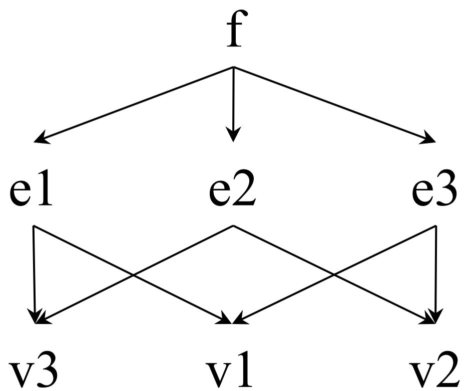
3-cell

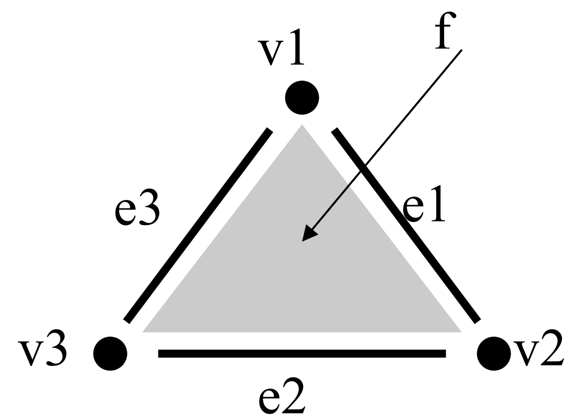


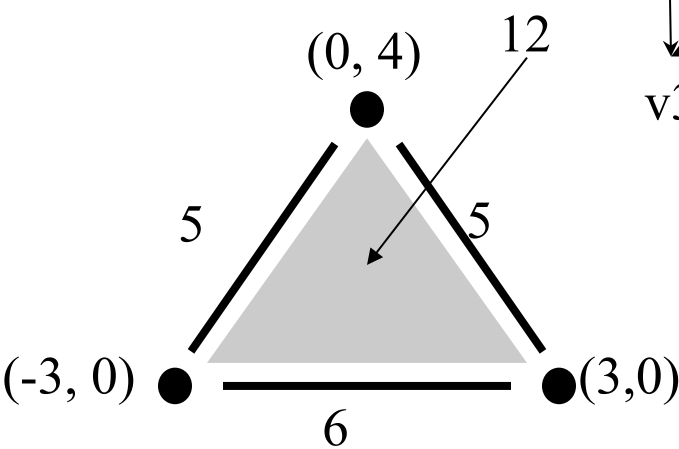
Abstract (Simplicial) Complex and (Simplicial) Chains

Incidence relationship and lattice of incidence:

- boundary(f) = {v1, v2, v3, e1, e2, e3}
- faces(f) = {e1, e2, e3}
- cofaces(v1) = {e1, e3}





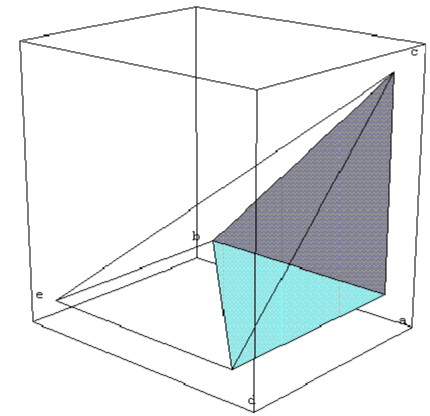


Topological chain

- coordinates with vertices
- lengths with edges
- area with f

$$\begin{pmatrix}0\\4\end{pmatrix}.v_1+\begin{pmatrix}3\\0\end{pmatrix}.v_2+\begin{pmatrix}-3\\0\end{pmatrix}.v_3+5.e_1+6.e_2+5.e_3+12.f$$

Topology ↔ simplicial complex↔ lattice



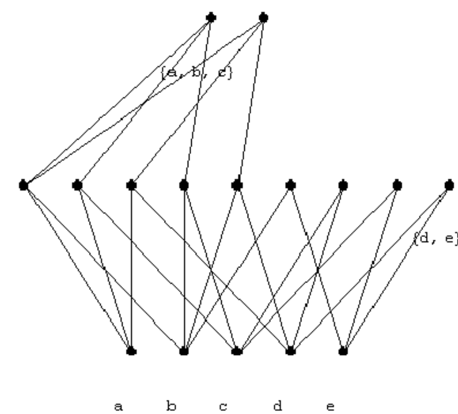
Topology (open and closed sets)

↓ simplices = Closed sets ↑

Simplicial Complex (set of sets closed by inclusion/intersection)

↓ simplices = Lattice cones ↑

Lattice (order relation: $\wedge$, $\vee$)



- Concise reformulation of classical approaches
- Extension

- Transformations
  - Functions defined by case on collections

    Each case (pattern) matches a sub-collection
  - Defining a rewriting relationship: ***topological rewriting***

trans T = **{**

$$pattern_1 \Rightarrow expression_1$$

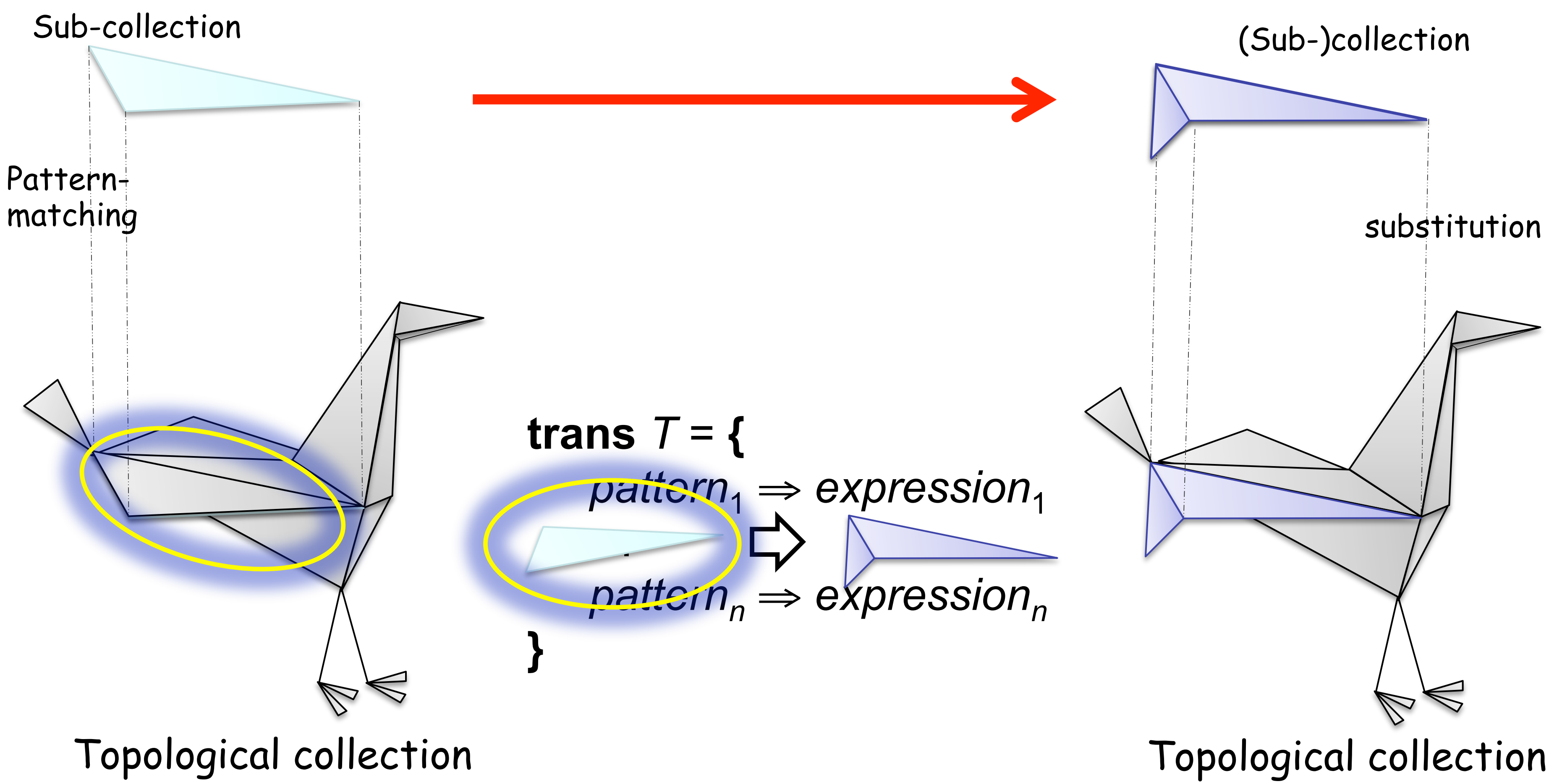
$$\ldots$$

$$pattern_n \Rightarrow expression_n$$

}

MGS Proposition

- Transformations



Topological rewriting = transformation

$1 + 2 \rightarrow \ldots$ (arithmetic) term rewriting

arithmetic operation

$a \, . \, b \rightarrow \ldots$ string rewriting (~ L systems)

string concatenation: « . » is a formal associative operation

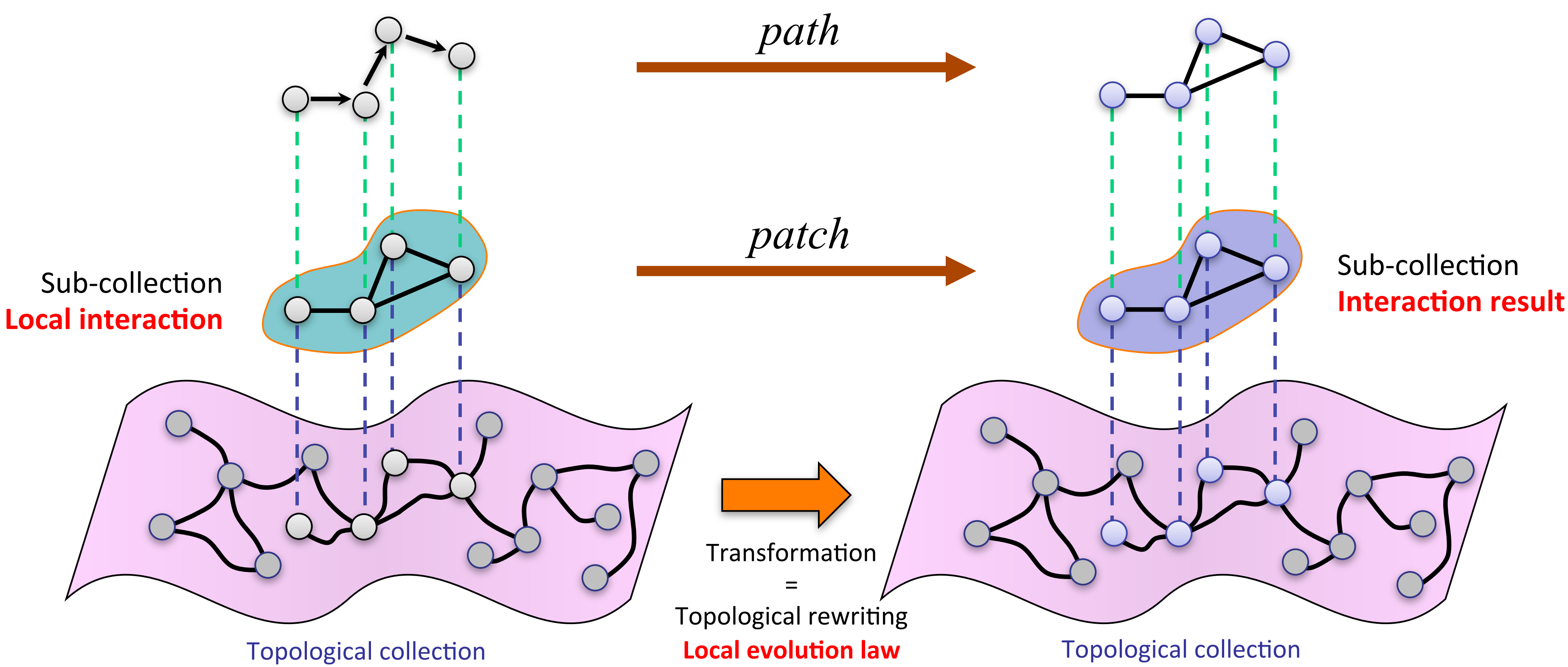
$2H + O \rightarrow H_2O$ multiset rewriting (~ chemistry)

multiset concatenation (= the chemical soup): « . » is AC

$v_1.\sigma_1 + v_2.\sigma_2 \rightarrow \ldots$ topological rewriting (MGS)

gluing cell in a cell complex: … (AC and algebraic machinery)

Transformation



Pattern matching : specifying a sub-collection of elements in interaction

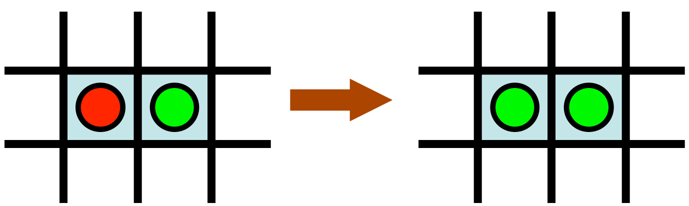
- *Path transformation* (path = sequence of neighbor elements)
  - Concise but limited expressiveness
- *Patch transformation* (arbitrary shape)
  - Longer but higher expressiveness

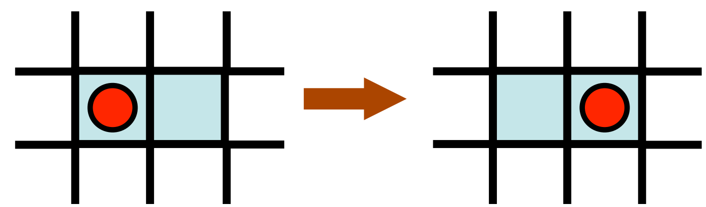
Example: Diffusion Limited Aggregation (DLA)

- Diffusion: some particles are randomly diffusing; others are fixed
- Aggregation: if a mobile particle meets a fixed one, it stays fixed

```
trans dla = {
      `mobile , `fixed  => `fixed, `fixed ;
      `mobile , <undef> => <undef>, `mobile
}
```

NEIGHBOR OF



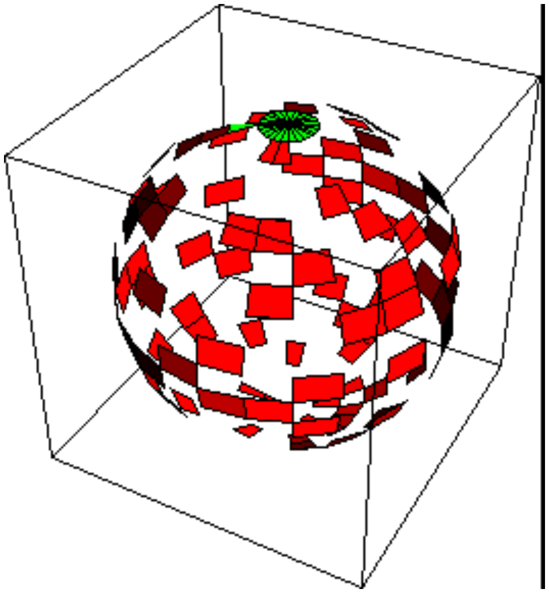


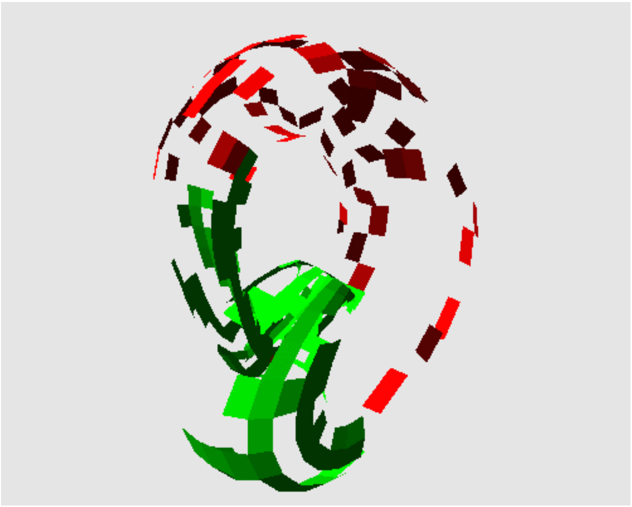
Example: Diffusion Limited Aggregation (DLA)

- Diffusion: some particles are randomly diffusing; others are fixed
- Aggregation: if a mobile particle meets a fixed one, it stays fixed

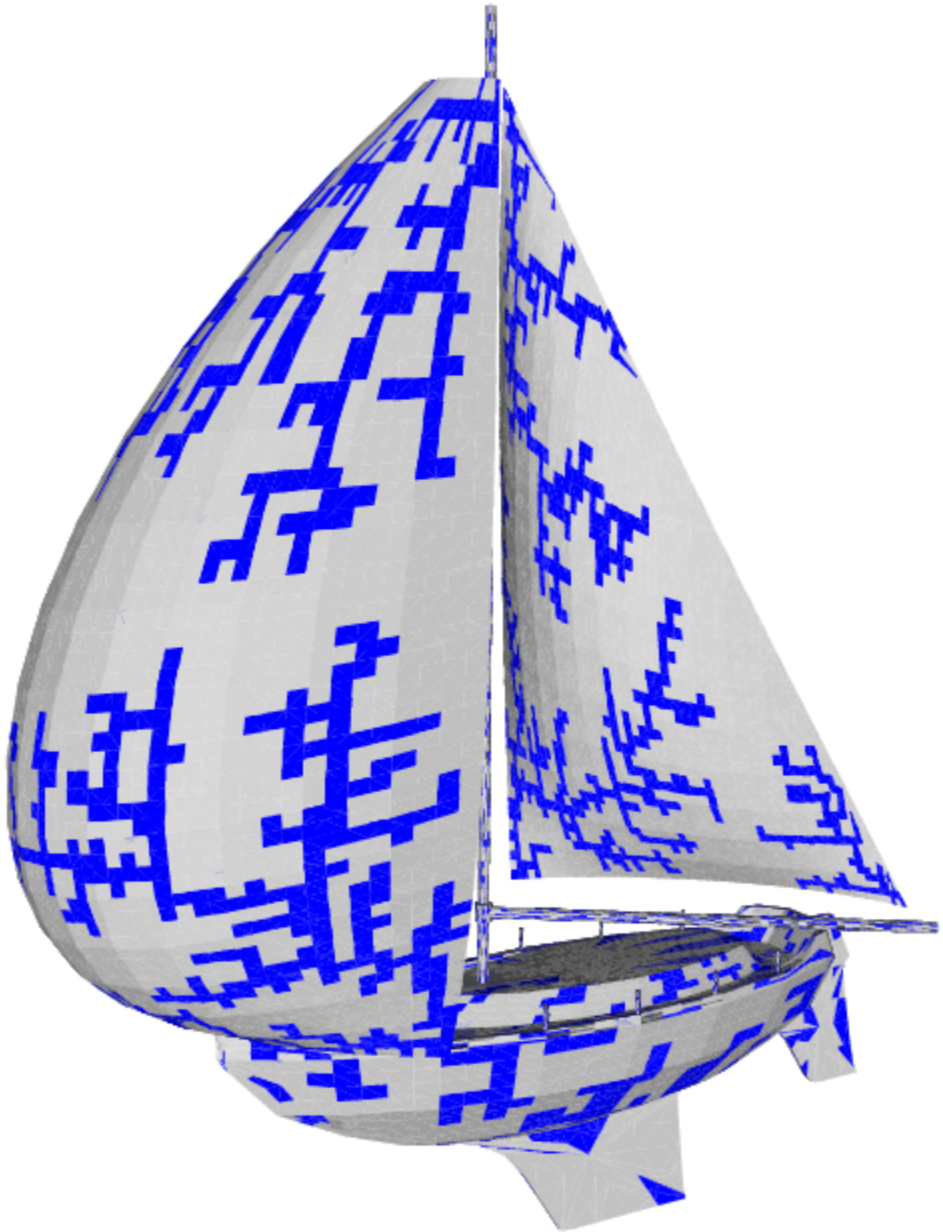
```
trans dla = {
     `mobile , `fixed  => `fixed, `fixed ;
     `mobile , <undef> => <undef>, `mobile
}
```

this transformation is an abstract process that can be applied to any kind of space



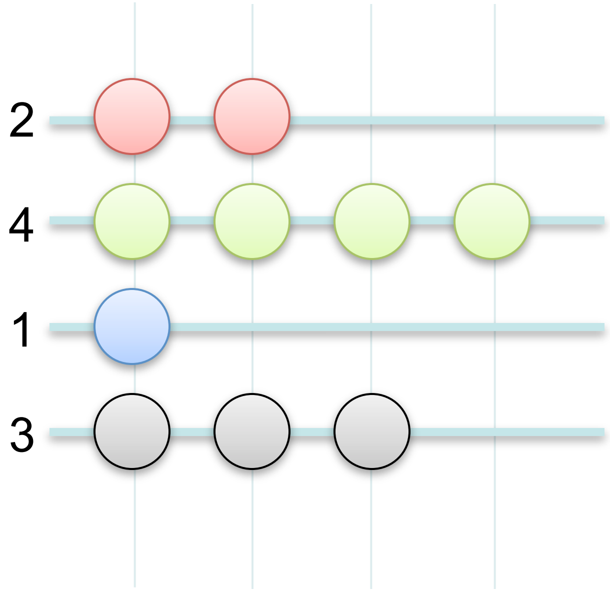


Polytypisme

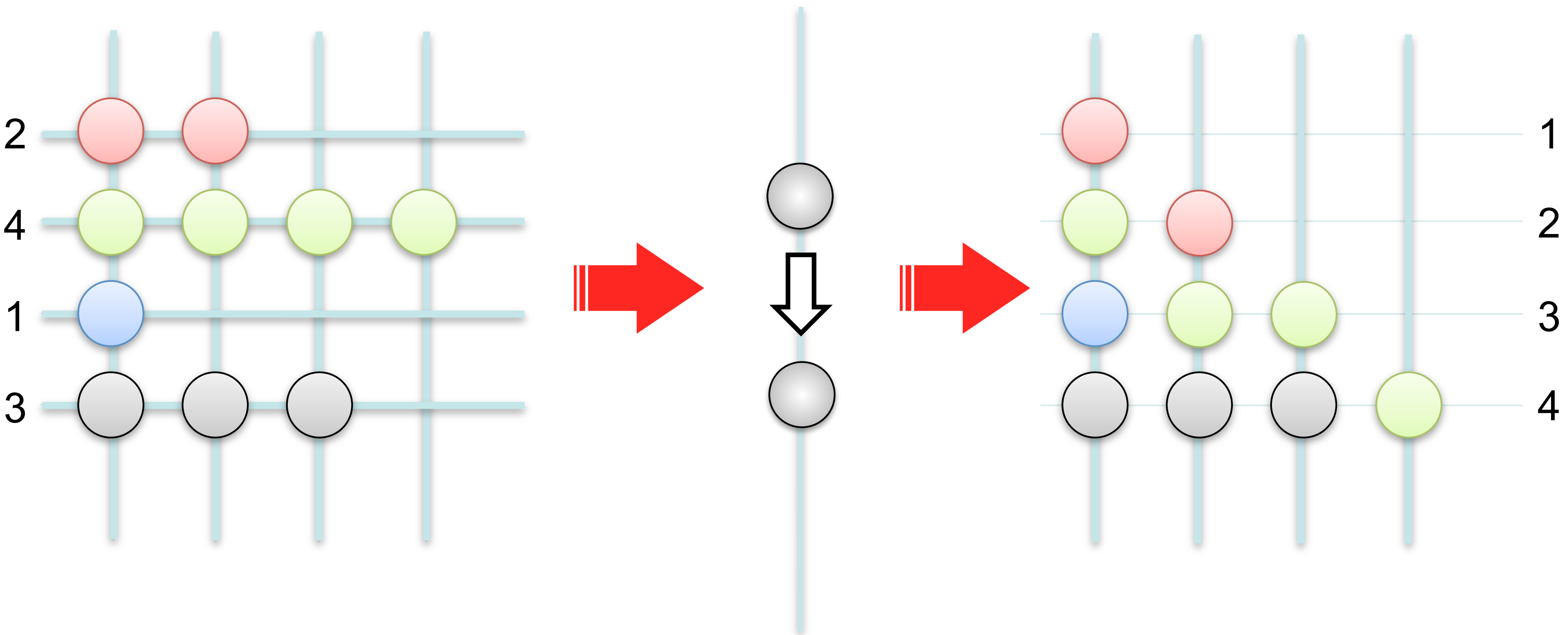


Algorithmic examples

Bead Sort



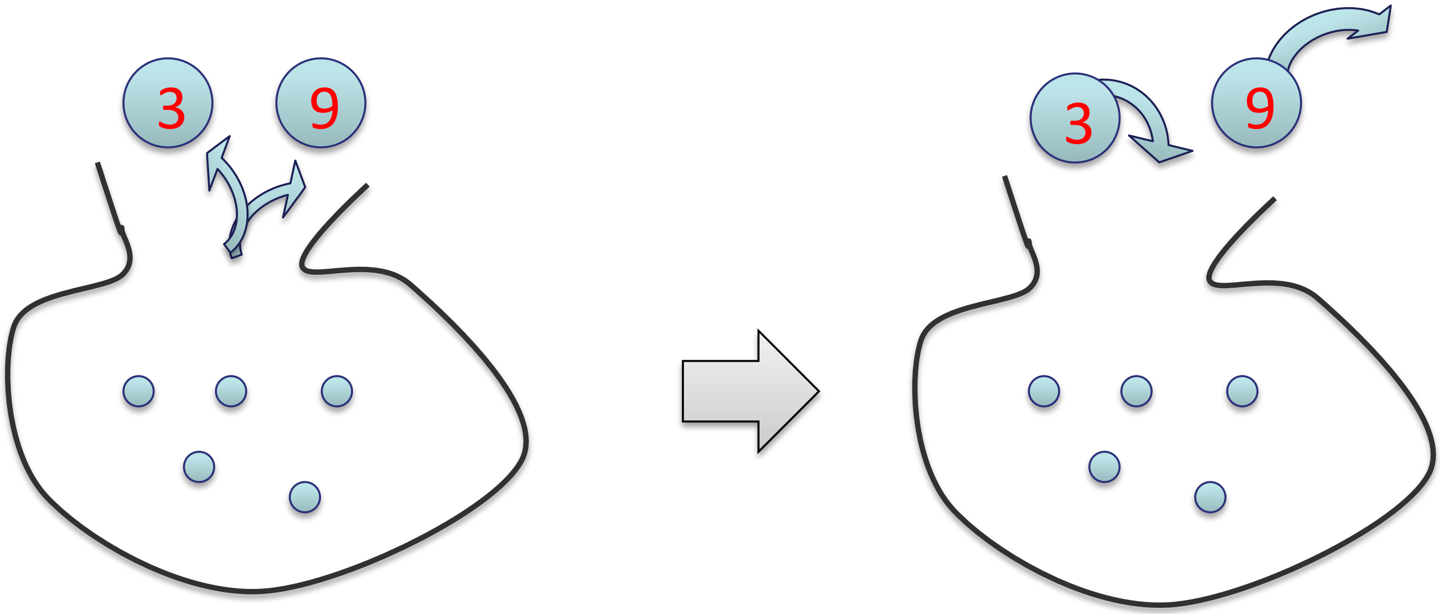
Bead Sort



Bead Sort

```
Gbf NEWS = < North, South, East, West;
             North+South=0, East+West=0>
trans dla = {
     `bead |south> `empty  => `empty, `bead ;
}
```

Eratosthene's Sieve

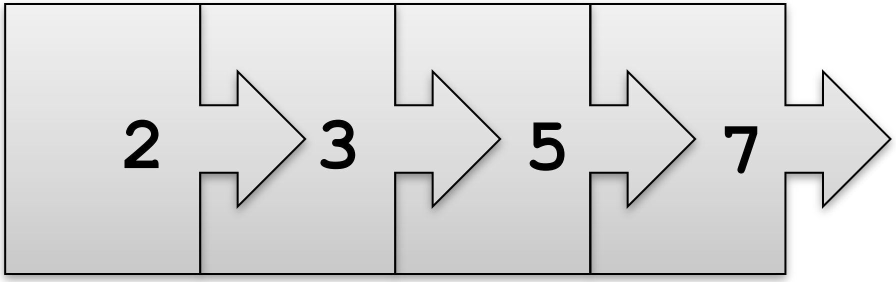


$$
\begin{aligned}
\texttt{trans}\ \mathit{Generate} &= \{x, \mathit{true}\} \ \texttt{=>}\ x, \{x+1, \mathit{true}\};\\
\texttt{trans}\ \mathit{Succed} &= \{x, \mathit{true}\} \ \texttt{=>}\ x;\\
\texttt{trans}\ \mathit{Eliminate} &= (x, y \,/\, y \,\texttt{mod}\, x = 0) \ \texttt{=>}\ x;
\end{aligned}
$$

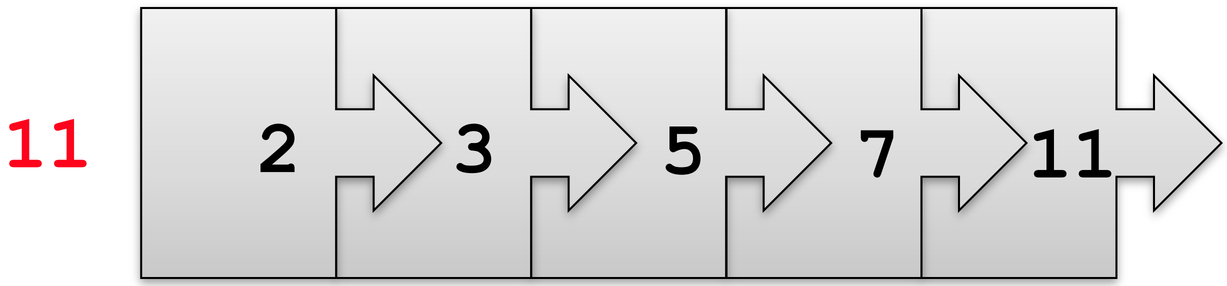
$$\mathit{Eliminate}[\texttt{fixrule}]\Big(\mathit{Succed}\big(\mathit{Generate}[N](\{2, \mathit{true}\}, \texttt{set}:())\big)\Big)$$

Eratosthene's Sieve

9

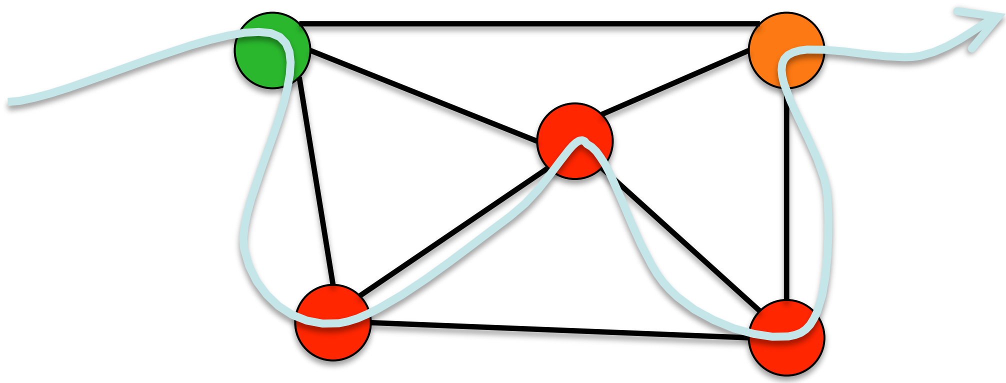


Eratosthene's Sieve

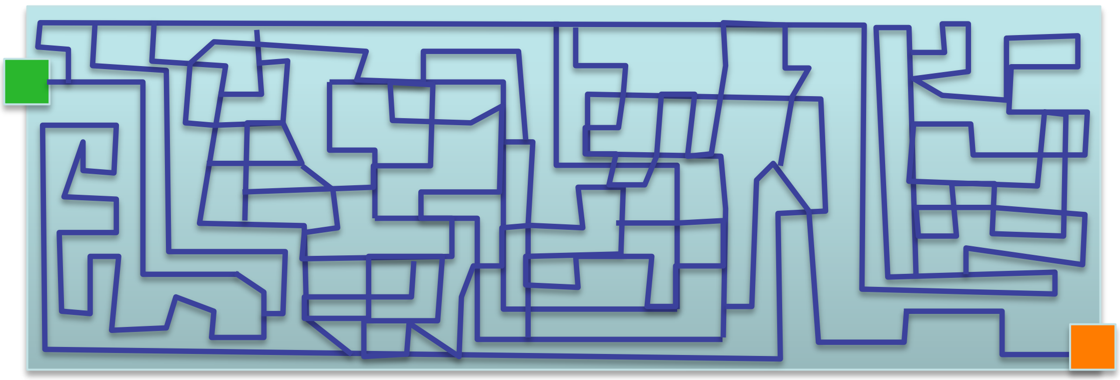


$$
\begin{aligned}
&\texttt{trans}\ \mathit{Eratos} = \{\\
&\quad \mathit{Genere1} \;=\; n : \mathit{integer} \,/\, \tilde{}\texttt{right}\ n \;=>\; n,\ \{\mathit{prime} = n\};\\
&\quad \mathit{Genere2} \;=\; n : \mathit{integer}, \{\mathit{prime}\ \texttt{as}\ x, \tilde{}\mathit{candidate}, \tilde{}\mathit{ok}\}\\
&\qquad\qquad =>\; n+1,\ \{\mathit{prime} = x,\ \mathit{candidate} = n\};\\
&\quad \mathit{Test1} \;=\; \{\mathit{prime}\ \texttt{as}\ x, \mathit{candidate}\ \texttt{as}\ y, \tilde{}\mathit{ok}\} \,/\, y \bmod x = 0 \;=>\; \{\mathit{prime} = x\};\\
&\quad \mathit{Test2} \;=\; \{\mathit{prime}\ \texttt{as}\ x, \mathit{candidate}\ \texttt{as}\ y, \tilde{}\mathit{ok}\} \,/\, y \bmod x <> 0\\
&\qquad\qquad =>\; \{\mathit{prime} = x, \mathit{ok} = y\};\\
&\quad \mathit{Next} \;=\; \{\mathit{prime}\ \texttt{as}\ x1, \mathit{ok}\ \texttt{as}\ y\}, \{\mathit{prime}\ \texttt{as}\ x2, \tilde{}\mathit{ok}, \tilde{}\mathit{candidate}\}\\
&\qquad\qquad =>\; \{\mathit{prime} = x1\}, \{\mathit{prime} = x2, \mathit{candidate} = y\};\\
&\quad \mathit{NextCreate} \;=\; \{\mathit{prime}\ \texttt{as}\ x, \mathit{ok}\ \texttt{as}\ y\}\ \texttt{as}\ s \,/\, \tilde{}\texttt{right}\ s\\
&\qquad\qquad =>\; \{\mathit{prime} = x\}, \{\mathit{prime} = y\};\\
&\}
\end{aligned}
$$

Hamiltonian path

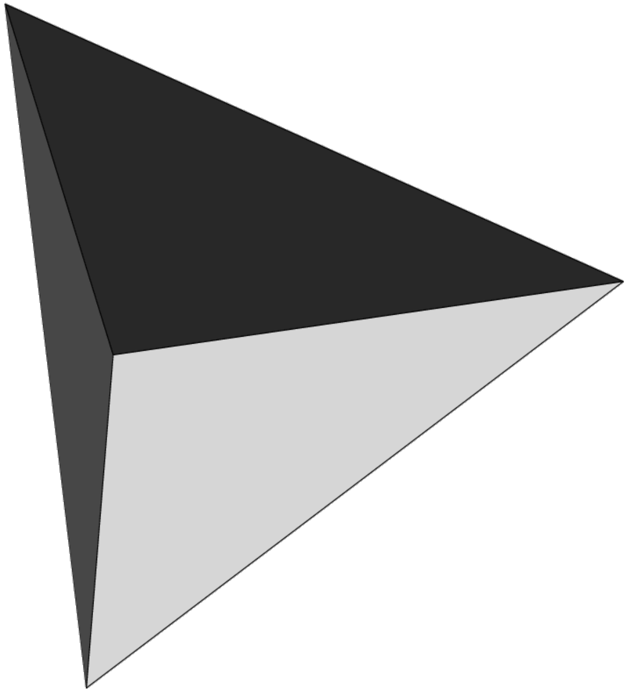


```
trans h_path = { `start , x* as p,`stop
                 / size(p) = n-2 => return p }
```

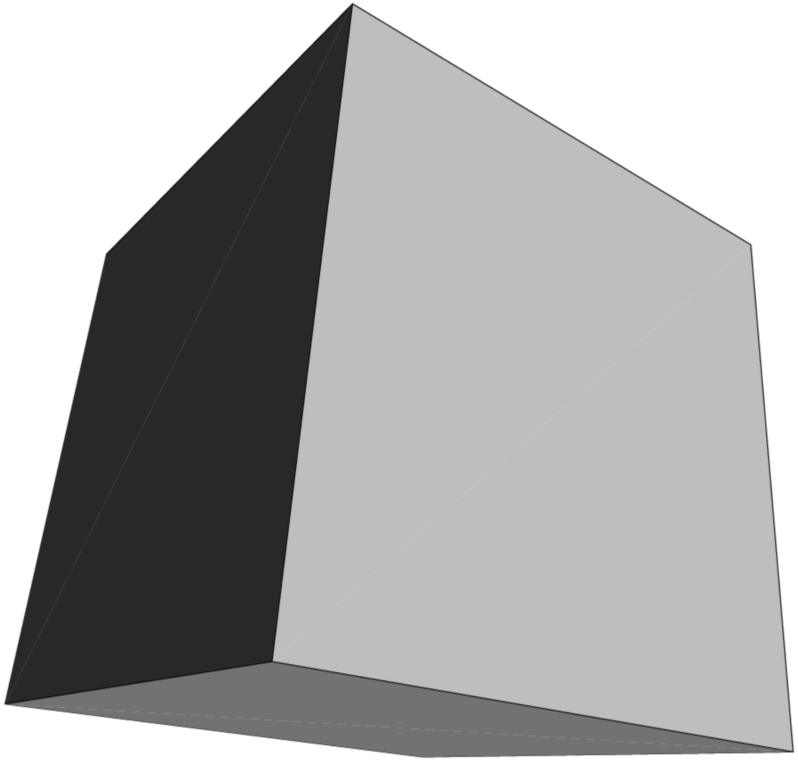


```
trans maze = { `input, c* as p,`output => return p }
```

Fractal construction by carving



Sierpinsky sponge (4 steps)



Menger sponge (2 steps)

Modeling of (bio)physical systems

- …
- Various models of Phage λ
- Sperm crawling
- Neurulation
- Prototyping a
  « synthetic multicellular bacteria »
- …

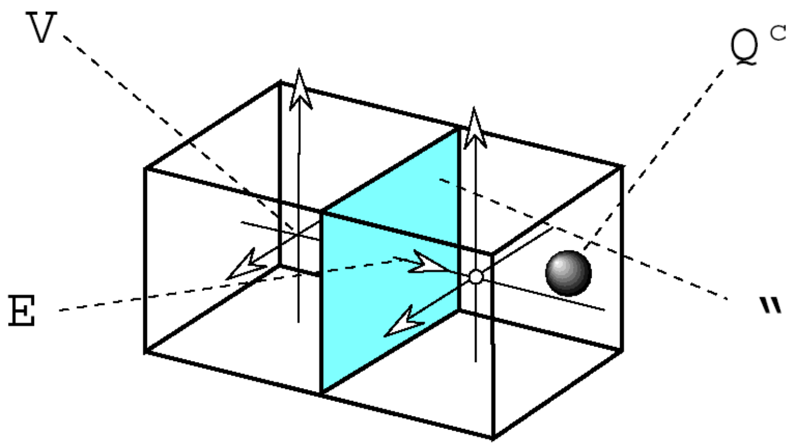
Higher dimensional objects for complex simulations

Example of electrostatic Gauss law [**Tonti** 74]

- Electric charge content ρ : dimension 3
- Electric flux $\boldsymbol{\Phi}$: dimension 2
- Law available on a arbitrary complex domain

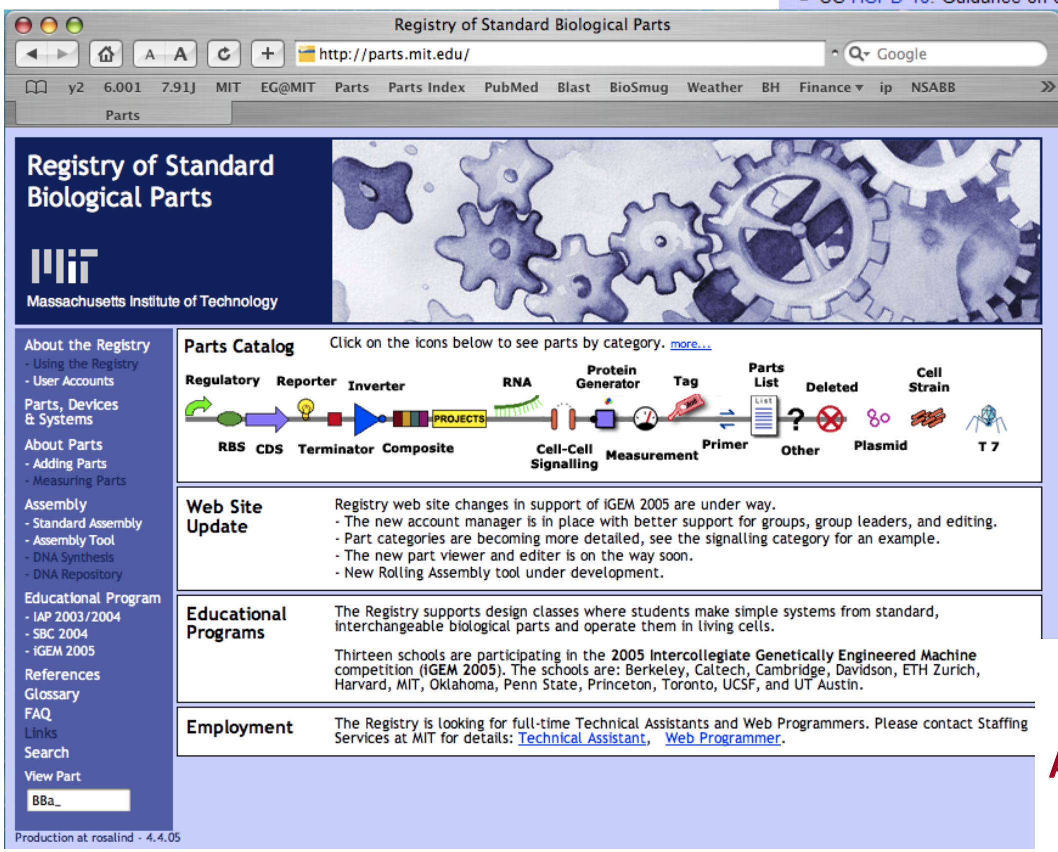
$$\phi = \oiint w \cdot dS = \frac{Q^c}{\varepsilon_0} = \iiint_{(V)} \frac{\rho}{\varepsilon_0} d\tau$$

electric field in space:
- V: electric potential (dim 0)
- E: voltage (dim 1)
- w: electric flux (dim 2)
- Qc: electric charge (dim 3)



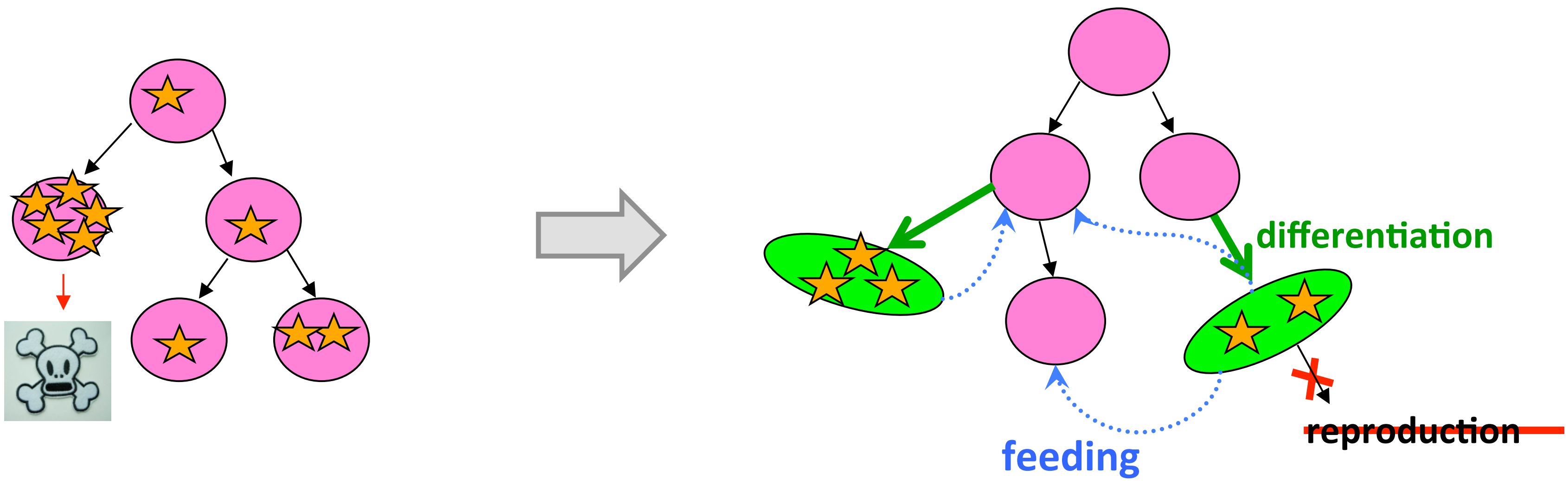
A Direct Discrete Formulation of Field Laws: The Cell Method

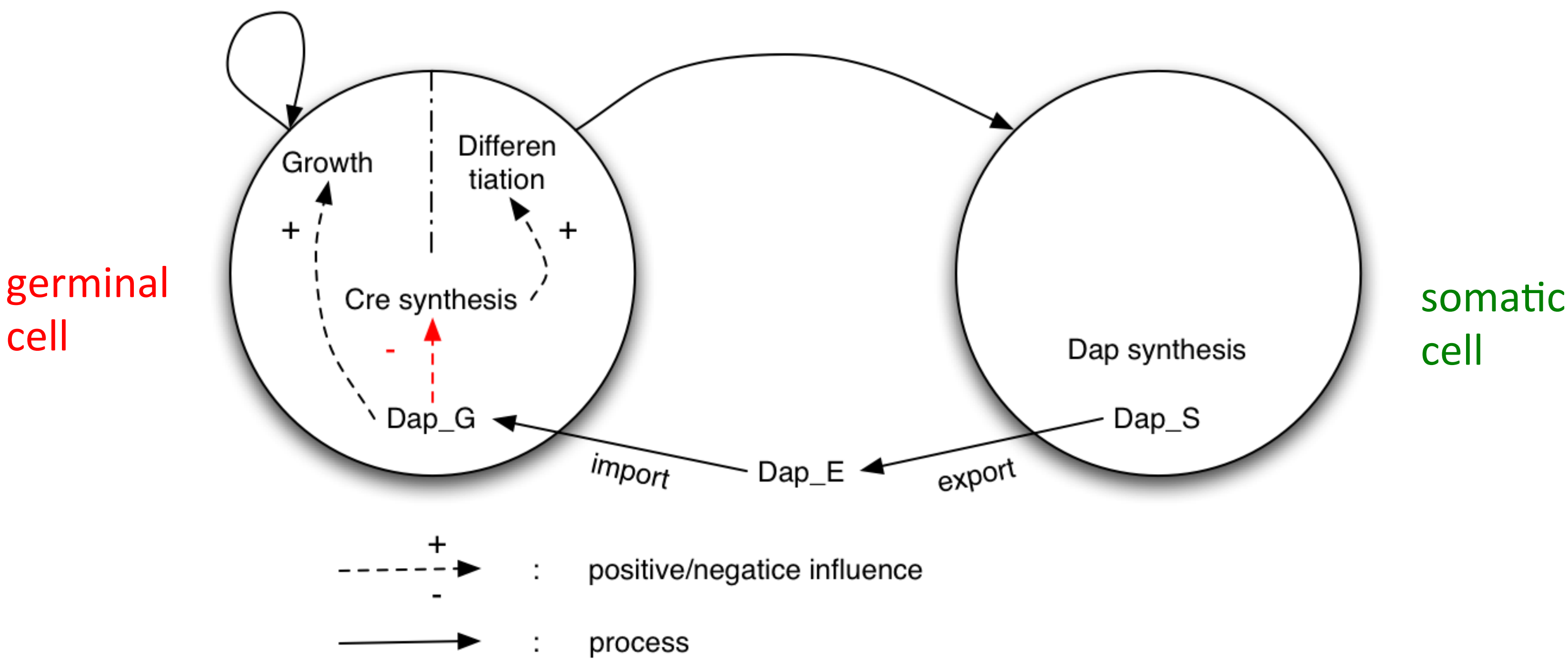
A Synthetic « Multicellular Bacterium »



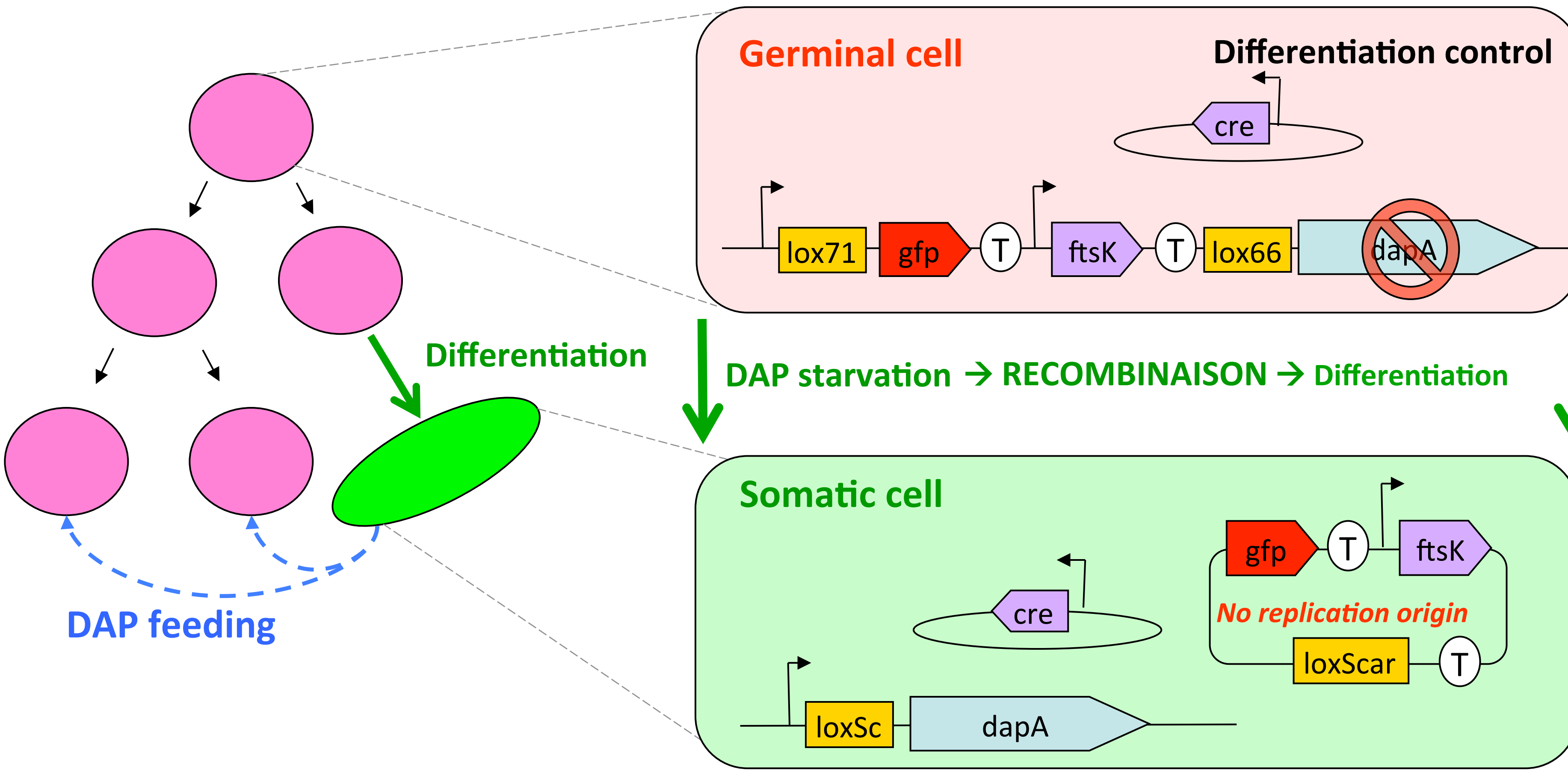
David Bikard, Thomas Landrain, David Puyraimond, Eimad Shotar, Gilles Vieira, Aurélien Rizk, David Guegan, Nicolas Chiaruttini, Thomas Clozel, Thomas Landrain

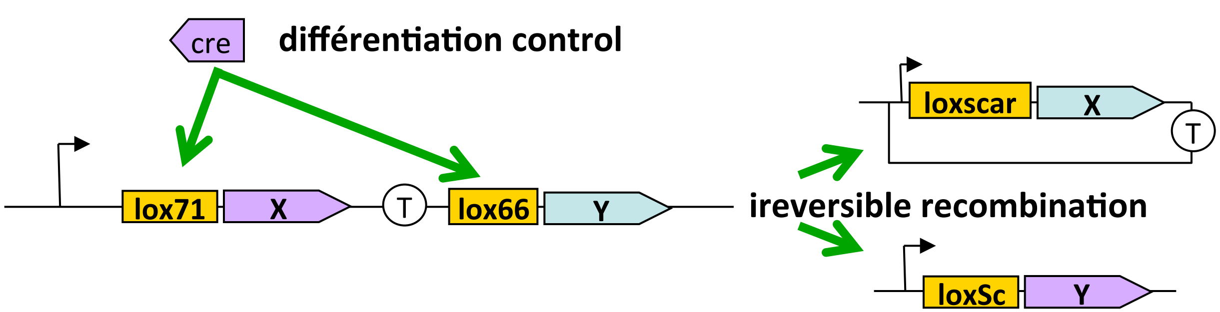
The Paris iGEM project: a « multicellular bacteria » to decouple growth and transgene expression

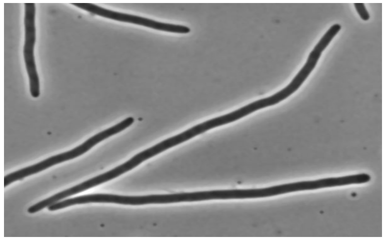




Implementation using BioBricks



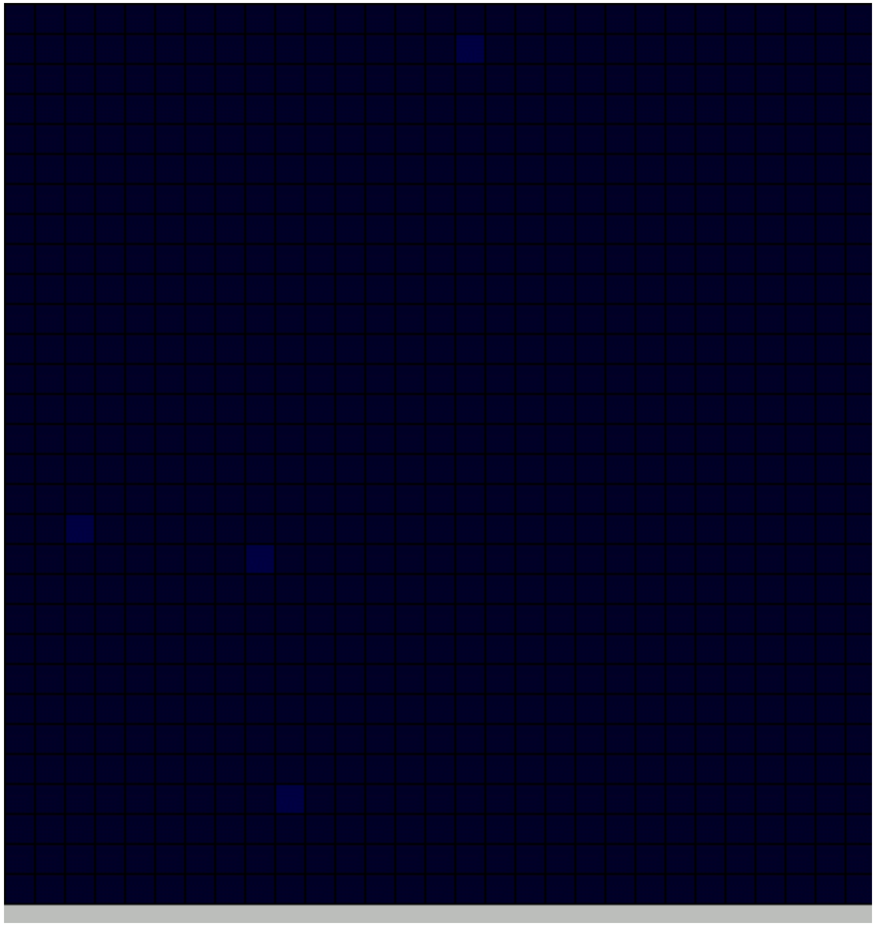




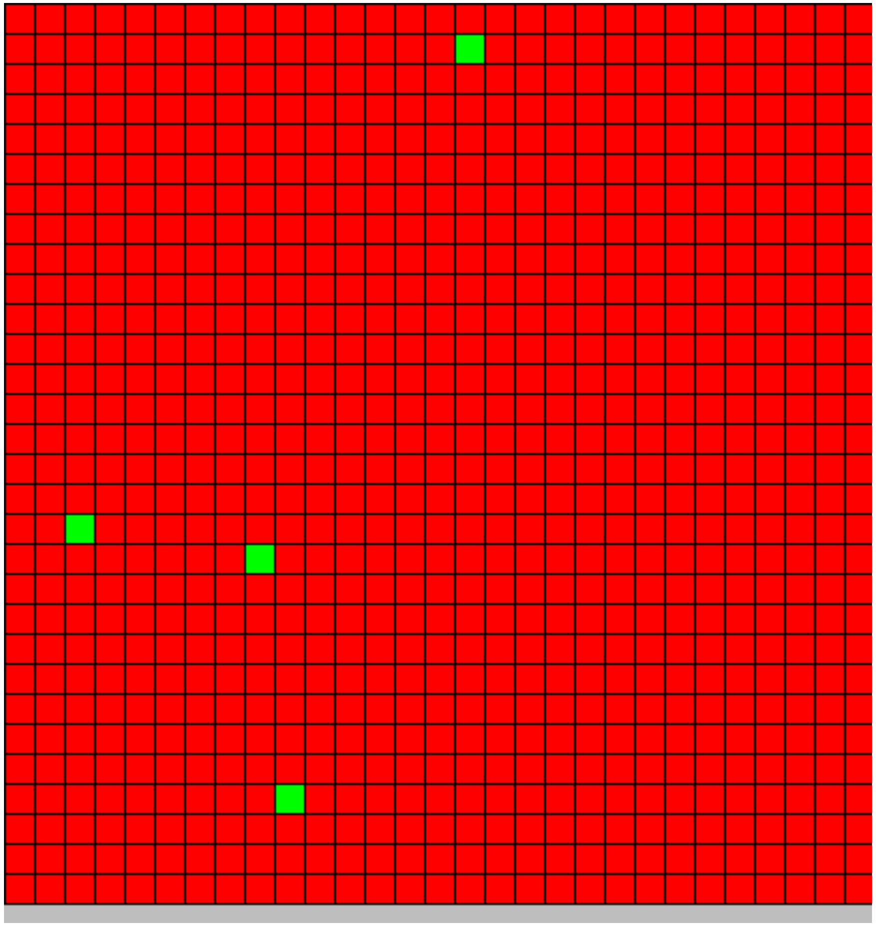
ftsK needed for cellular division

Proof of Concept: Simulation to answer 4 questions

- **How does differentiation induces feeding?** (proof of concept)
  cellular automaton (in MGS)



diffusion of DAP



somatic and germ cell

Proof of Concept: Simulation to answer 4 questions

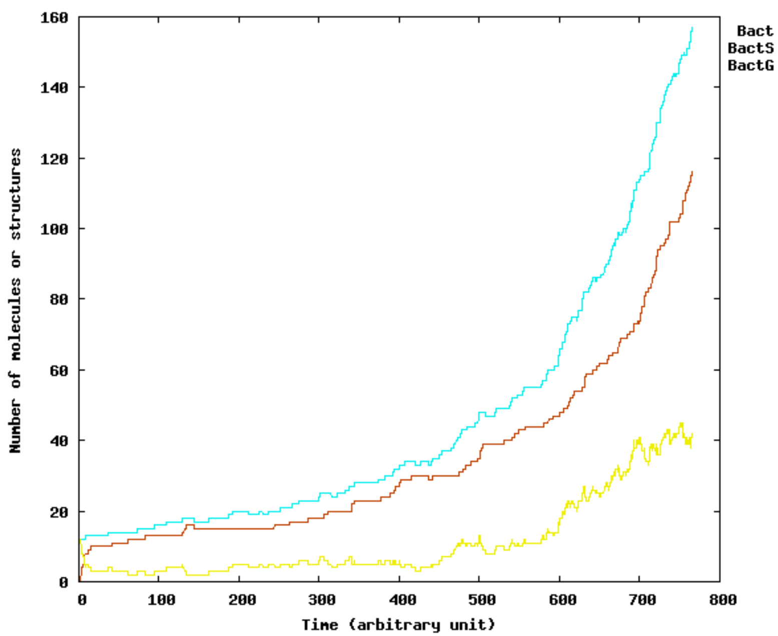
- **How does differentiation induces feeding?** (proof of concept)
  cellular automaton (in MGS)
- **How do spatial organization and distribution evolve?**
  agents based system (in MGS)

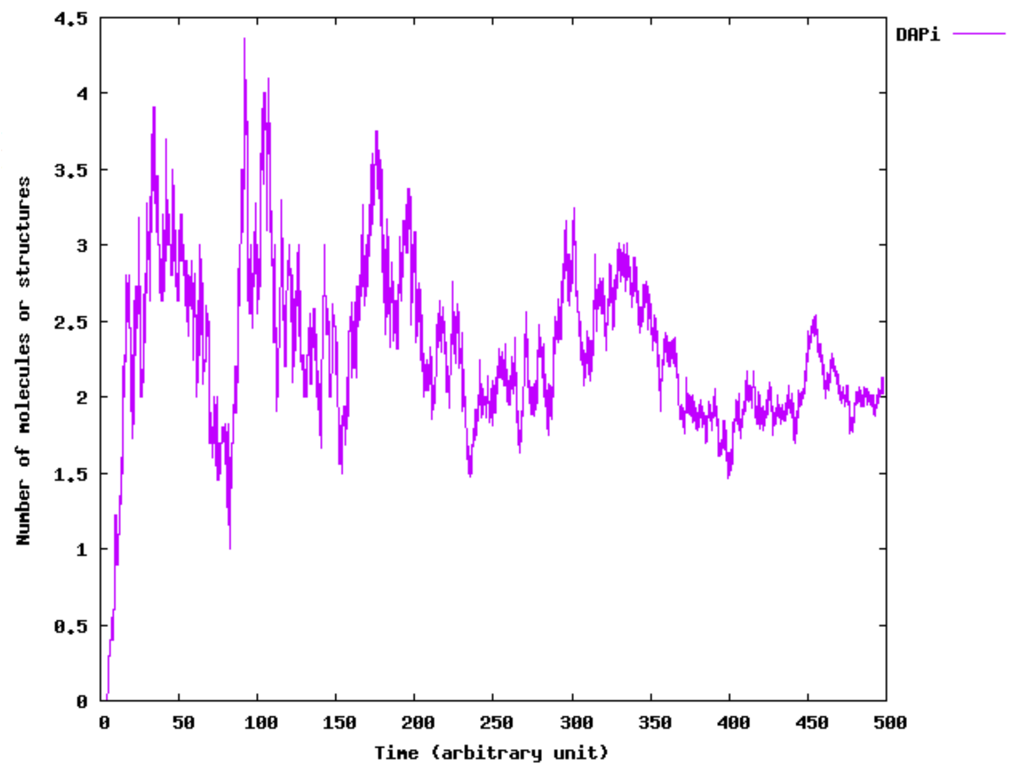
Proof of Concept: Simulation to answer 4 questions

- **How does differentiation induces feeding?** (proof of concept)
  cellular automaton (in MGS)
- **How do spatial organization and distribution evolve?**
  agents based system (in MGS)
- **How robust and tunable is the model?**
  ODE kinetics (matlab)

Proof of Concept: Simulation to answer 4 questions

- **How does differentiation induces feeding?** (proof of concept)
  cellular automaton (in MGS)
- **How do spatial organization and distribution evolve?**
  agents based system (in MGS)
- **How robust and tunable is the model?**
  ODE kinetics
- **How sensitive is the system to noise?**
  Gillespie based simulation (in MGS)





Conclusions and Perspectives

MGS drawbacks and successes

Success

- Polytypisme is good
- Rule application strategies are good
- Patterns/rules are expressive and usually concise
- Clean semantics

Shortcomings

- Rules may be heavy (e.g. 100 variables for the fractal sponge)
  graphical drawing of rules
  look for better notations (e.g. path pattern)
- Efficiency
  well…
- Implicit methods (solvers) are hairy
  use explicit ones

Perspectives

- An intrinsic complexity theory
  e.g., w.r.t. interactions
- A logic of spatial interactions
- Relationships to physics
  a discrete differential calculus (cf. PhysicaD 08)
- Internalizing time
- Implementation
  pattern-matching compilation and optimization, specific abstract combinatorial complex, parallelism
- Non standard applications
  e.g., in knowledge representation
  or in music analysis (Louis Bigo talk)

A topological manifesto

Spatial computing proposes to celebrate
corporeality of data rather than trying to deny it.
Simon Greenworld (MIT medialab)

- The logical approach in computer science
  computation = deduction
  (the Curry-Howard isomorphism)

- Other paradigms can be fruitfull : topology
  computation = moving in a space

- Try to perceive space (and time) in programs
  (rather than logical operations)
  purposes: *technical, heuristic, didactical*

http://scw11.spatial-computing.org

MGS

Call for Papers: Spatial Computing Workshop 2011

at 5^{th} IEEE International Conference on Self-Adaptive and Self-Organizing Systems
Ann Arbor, Michigan, USA, October 3, 2011

Organizers: Jacob Beal (BBN Technologies, USA) Stefan Dulman (Delft University, the Netherlands) Olivier Michel (University Paris-Est Créteil, France) Antoine Spicher (University Paris-Est Créteil, France)

Submission Deadline: July 4^{th}, 2011

Many self-organizing or self-adaptive systems are "spatial computers" – collections of local computational devices distributed through physical space, in which:

- the difficulty of moving information between any two devices is strongly dependent on the distance between them, and
- the "functional goals" of the system are generally defined in terms of the system's spatial structure.

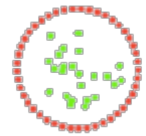
ystems that can be viewed as spatial computers are abundant, both natural and man-made. For example, in wireless sensor networks and nimal or robot swarms, inter-agent communication network topologies are determined by the distance between devices, while the agent ollectives as a whole solve spatially-defined problems like "analyze and react to spatial temperature variance" or "surround and destroy n enemy."

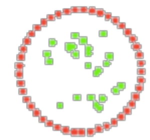
Similarly, in reconfigurable microchip platforms, moving data between adjacent logic blocks is much faster than moving it across the hip, which in turn favors problems with spatial structure like stream processing. In biological embryos, each developing cell's behavior controlled only by its local chemical and physical environment, but the eventual structure of the organism is a global property of the

Thanks

- **Antoine Spicher**
- **Olivier Michel**

http://mgs.spatial-computing.org





- **PhD and other students**

  Louis Bigo
  J. Cohen, P. Barbier de Reuille,
  E. Delsinne, V. Larue, F. Letierce, B. Calvez,
  F. Thonerieux, D. Boussié *and the others…*

- **Past and presents Collaborations**
  - A. Lesne (IHES, stochastic simulation)
  - P. Prusinkiewicz (UoC, declarative modeling)
  - P. Barbier de Reuille (meristeme model)
  - C. Godin (CIRAD, biological modeling)
  - H. Berry (INRIA, stochastic simulation)
  - G. Malcolm (Liverpool, rewriting)
  - J.-P. Banâtre (IRISA, programming)
  - P. Fradet (Inria Alpes, programming)
  - F. Delaplace (IBISC, synthetic biology)
  - P. Dittrich (Jena, chemical organization)
  - F. Gruau (LRI, language and hardware)
  - P. Liehnard (Poitier, CAD, Gmap and quasi-manifold)