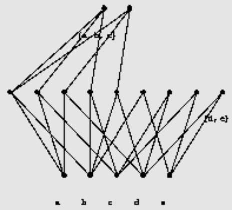
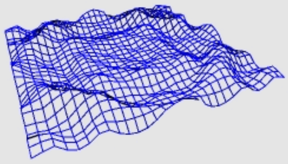
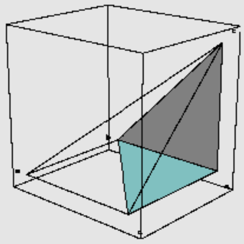
Space in Correlations
& Correlations in Space



Olivier Michel
Jean-Louis Giavitto





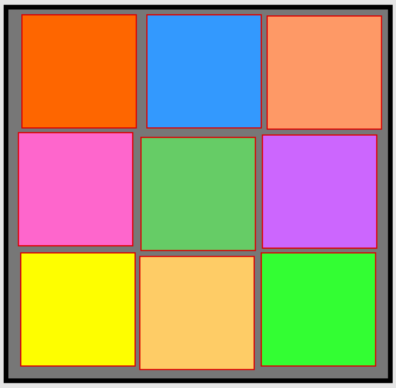
ibiSc
Informatique, Biologie Integrative et Systèmes Complexes (IBISC)
FRE 3190 CNRS - Université d'Evry - Genopole

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

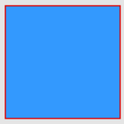
- Motivations
- The topological structure of interacting parts
- Q-analysis
- Data structure as space (and control structure as path): the MGS programming language
- From global to local: rules and differential operators
- Examples
- Conclusion and perspectives

The topological structure of the interactions of an aggregate dynamical system

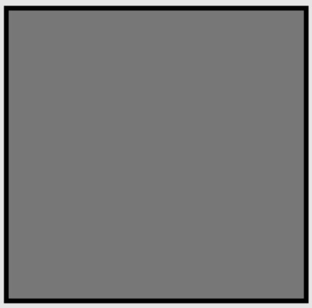
the state of a system



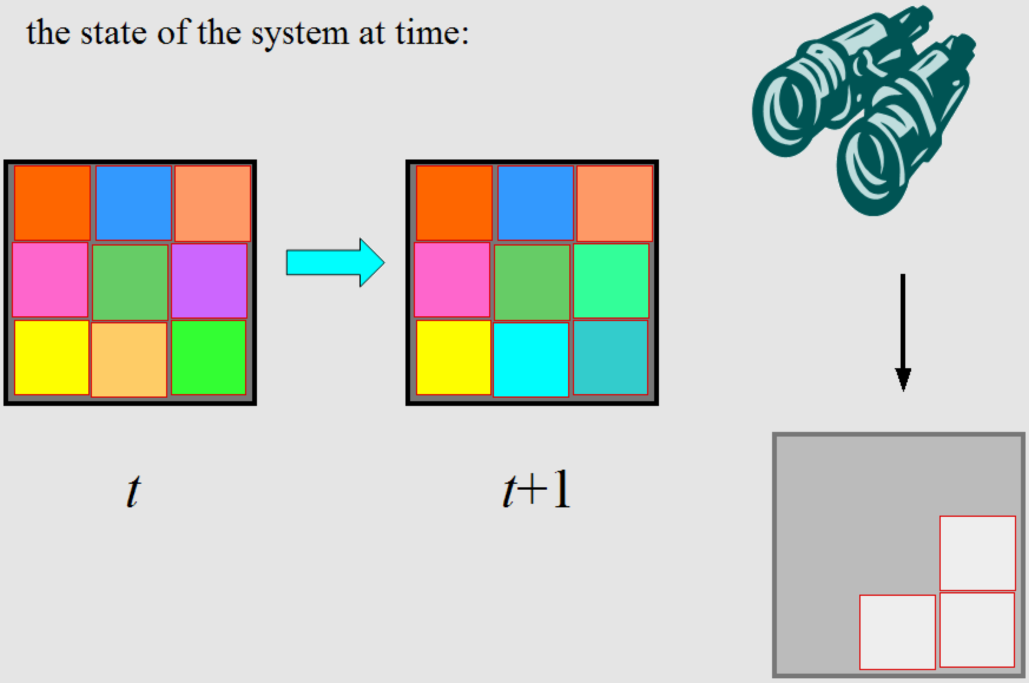
the state of a *sub*-system



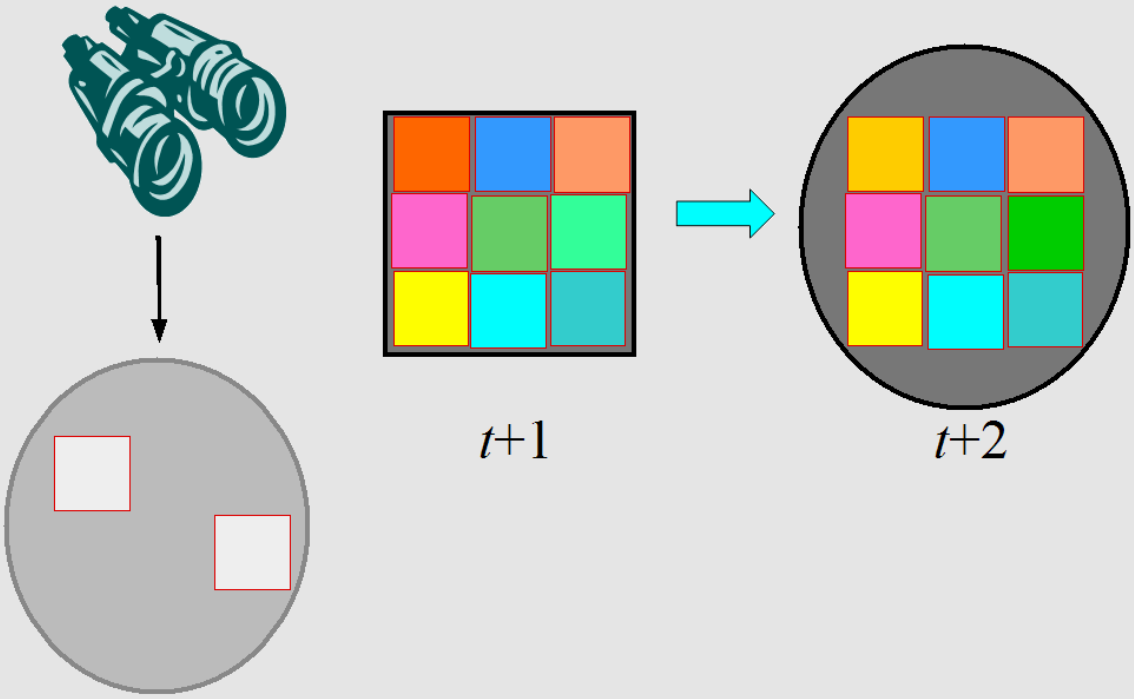
the structure of the composition:
vector, set, matrix, stack, field, ...



the state of the system at time:

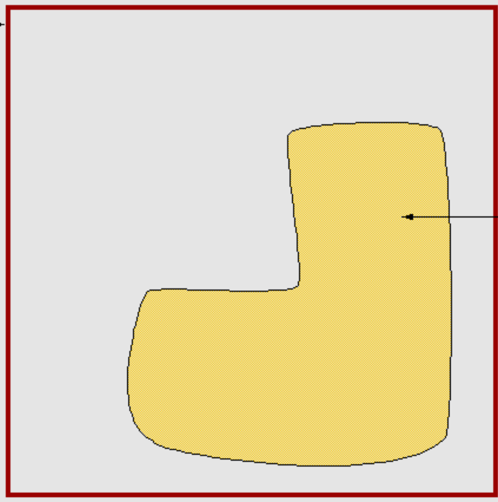


the state of the system at time:



Decompose a system into sub-systems following the elements in interaction

a system
in some state

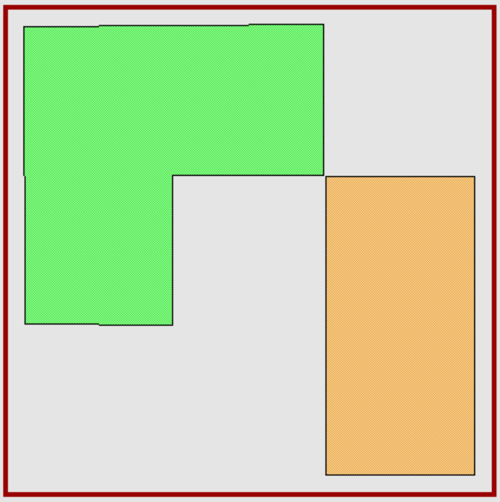


Part of a system that evolves. These elements are ***correlated wrt the evolution***

Can be identified by comparaison with the previous global *state*

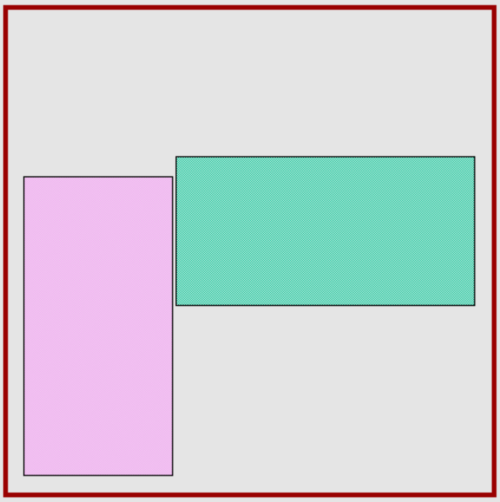
Decompose a system into sub-systems following the elements in interaction

$t = 1$



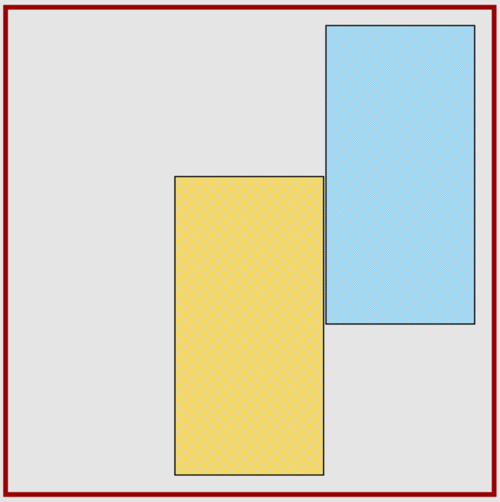
Decompose a system into sub-systems following the elements in interaction

$t = 2$

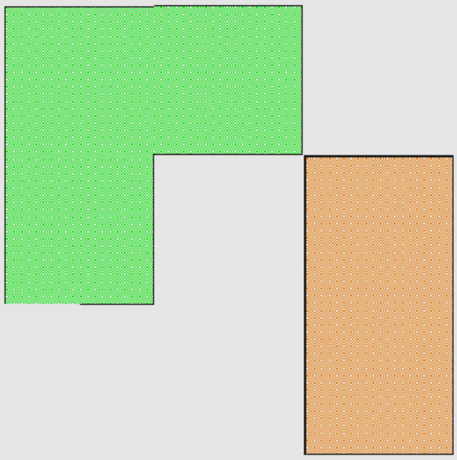


Decompose a system into sub-systems following the elements in interaction

$t = 3$

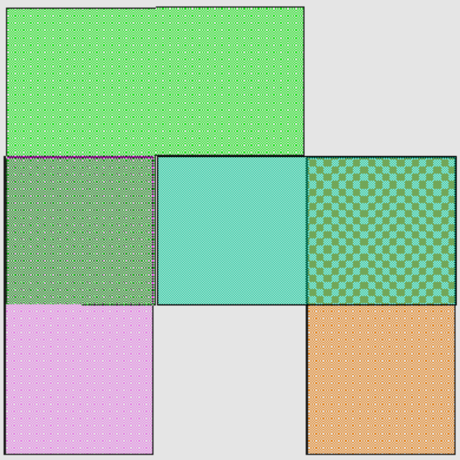


Interaction structure of a system: Interaction structure of a system (5)



...

Interaction structure of a system: Interaction structure of a system (5)

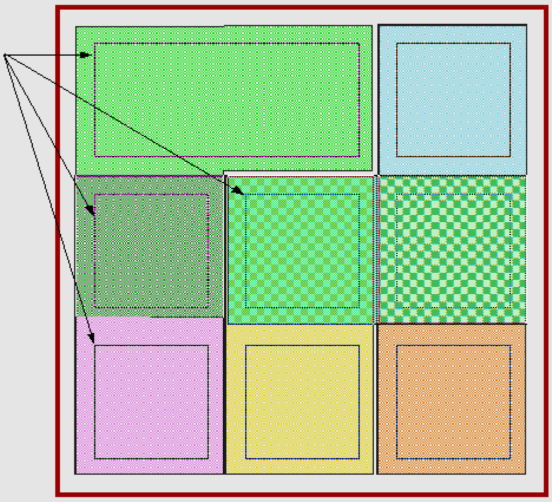


...

Decompose a system into sub-systems following the elements in interaction

elementary parts
of the system

each elementary part has its own local *state*



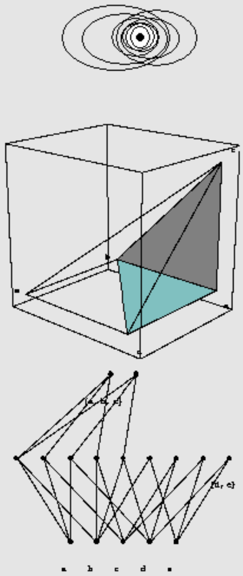
Decompose a system into sub-systems following the elements in interaction

the inclusion structure between the elementary and interacting parts is
a lattice

⇕

a (simplicial) complex
is a better (topological) equivalent representation

Interaction structure of a system: Topology <--> simplicial complex <--> lattice



topology (open and closed sets)

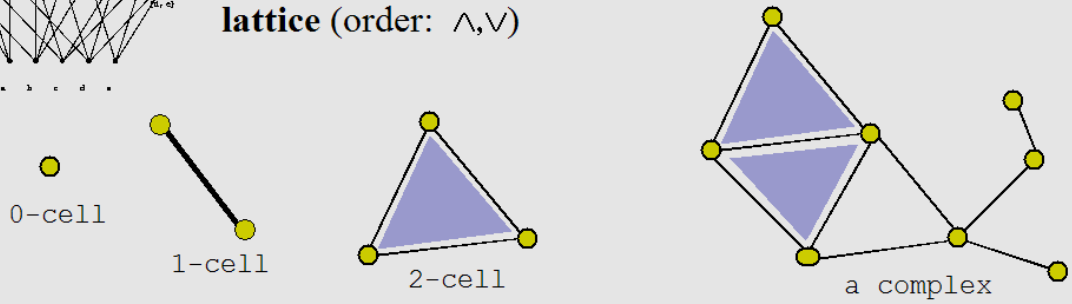
simplex = closed set

simplicial complex (set of subsets closed by intersection)

simplices = cones of the lattice

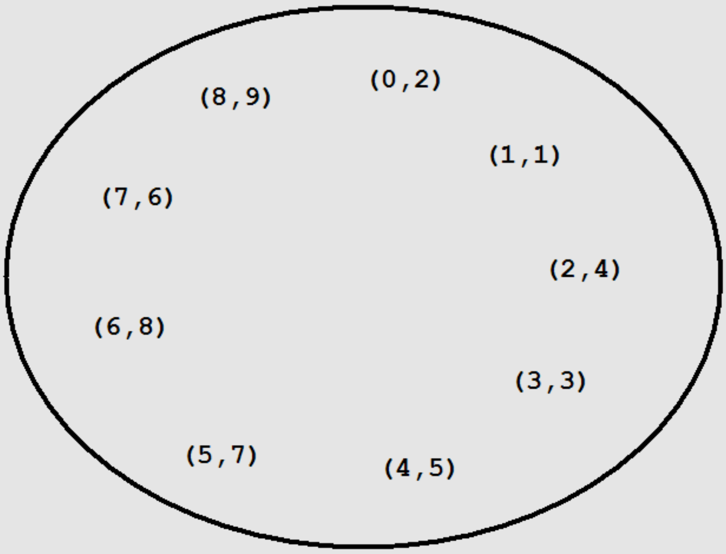
$(\{\text{simplices}\}, \subseteq)$

lattice (order: $\wedge, \vee$)



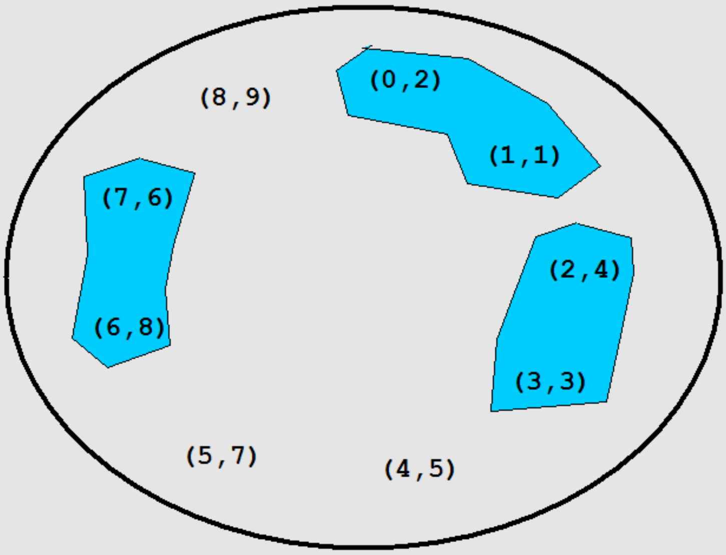
Interaction structure of a system: A motivating example

t = 0



...

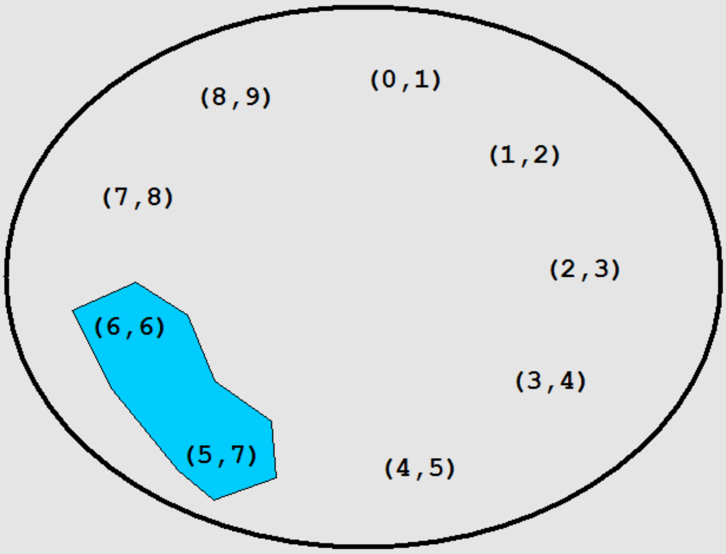
Interaction structure of a system: A motivating example



...

Interaction structure of a system: A motivating example

t = 1



...

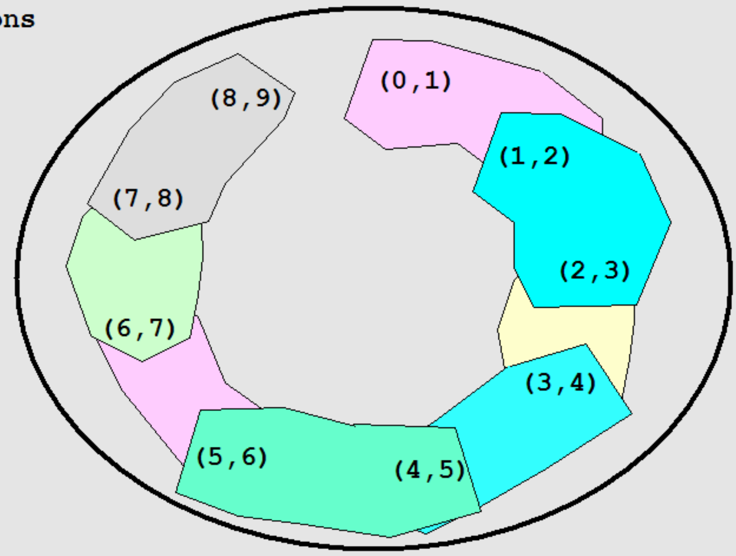
Interaction structure of a system: A motivating example

t = 2

...

Interaction structure of a system: A motivating example

all potential interactions



$$(0, x_1) — (1, x_2) — (2, x_3) — (3, x_4) — (4, x_5) — (5, x_6) — (6, x_7) — (7, x_8) — (8, x_9)$$

section 2/10 page 15/16 Correlation Vielsalm'08 – MGS / IBISC FRE 3190 CNRS - Université d'Evry - Geonopole Quit Menu

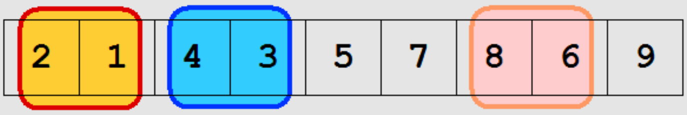
Interaction structure of a system: a concrete example

$t = 0$

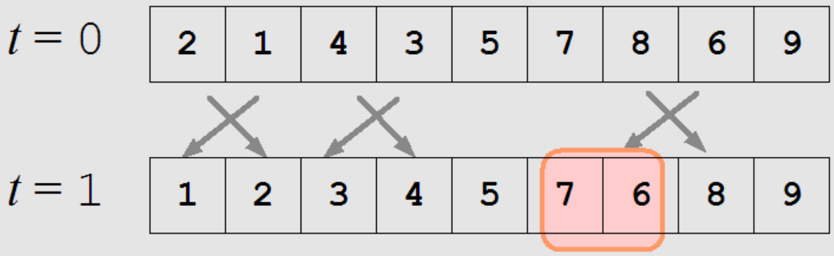
Example of an abstract process: sorting a sequence of number

Interaction structure of a system: a concrete example

$t = 0$



Example of an abstract process: sorting a sequence of number



Example of an abstract process: sorting a sequence of number

...

Interaction structure of a system: a concrete example

$t = 0$	2	1	4	3	5	7	8	6	9
$t = 1$	1	2	3	4	5	7	6	8	9
$t = 2$	1	2	3	4	5	6	7	8	9

Bubble-sort is a process where:
- the state of the system is a sequence of numbers
- an interacting part in the system is a pair of adjacent decreasing numbers
- the transformation of an interacting couple exchanges the couple's elements
- the topology of the interacting parts is build upon the topology of the sequence

or

- the topology of the sequence can be recovered from the possible element's swap

section 2/10 page 16/16 Correlation Vielsalm'08 – **MGS** / IBISC FRE 3190 CNRS - Université d'Evry - Geonopole Quit Menu

The big picture

A research program

1) Analyse: structural properties of the topological structure of interacting parts: Q-analysis

2) Simulation: program the evolution of arbitrary interacting parts

3) *Develop applications*

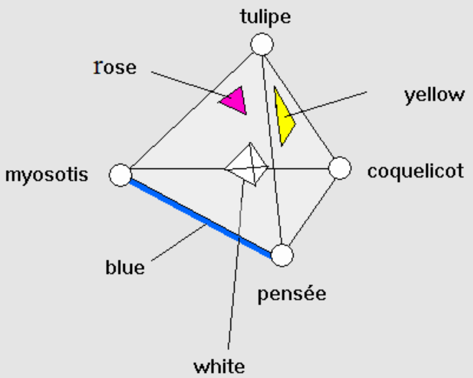
Q-analysis
(*Atkins* 1974)

Binary relationship

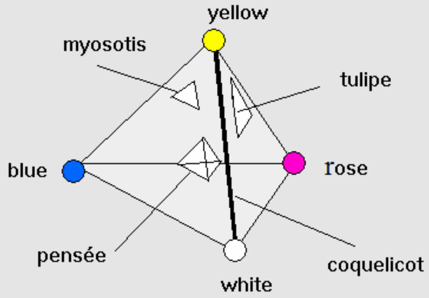
R ⊂ Fower × Color

Flower = {tulipe, coquelicot, pensée, myosotis}
Color = {rose, jaune, bleu, blanc}

	tulipe	coquelicot	pensée	myosotis
rose	1	0	1	1
yellow	1	1	1	0
blue	0	0	1	1
white	1	1	1	1



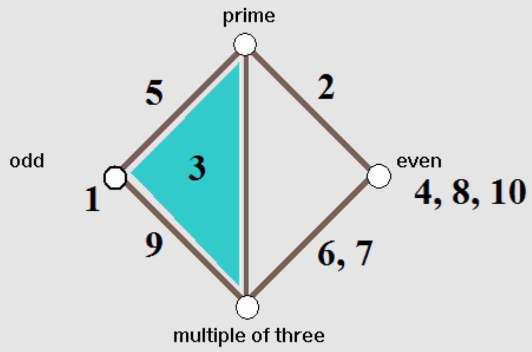
CS representation



dual CS representation

λ ⊂ Objects × Predicates : (o,p)∈ λ iff p(o)

Objects = {1,2,3, ..., 10}
Predicates = {even, odd, prime, multiple-of-three}



The simplices s_0 and s_r are ***q*-connected** if there exists a sequence of simplices $\{s_i\}$, i = 0, 1, 2, ..., r such that

1. s_i shares m vertices with s_1 .
2. s_{r-1} shares n vertices with s_r .
3. The face between s_k and s_{k+1} is of dimension d_k.
4. $\boldsymbol{q} = \min\{d_0, ..., d_{r-1}\} + 1$

q-connectedness is reflexive, symmetric and transitive

Equivalence classes represent ***coupling:*** elements that "act together" or "share properties". The parameter $\boldsymbol{q}$ is the force of the coupling.

Example of applications: [Casti 1992]

Various possible predicates to analyse the system dynamics

- {P_t = has changed at time t } (familly of binary predicates)
- $P(x,y)$ = x changed at time t entail y changes at time t+1
- ...
- Regulatory networks given by their interaction graph.
  Application to their visualisations:
  - Successor or predecessor relation (or their union)
  - Rips complex:
    . for each p in **R**, genes g and g' are in the same simplex if $d(g, g') < p$
    . we take d has the Hamming distance in the incidence matrix
    . application to the *Yeast genes regulatory network*

Partition	q = 3	q = 4	Function
1	*yol052ca, ctt1(ygr088w), hsp12(yfl014w), tps2(ydr074w), cyt1(yor065w), cyc1(yjr048w), cox6(yhr051w), sod2(yhr008c)*	-	genes of the respiratory chain, protection against oxidative damage and other stresses
2	*ste6(ykl209c), ste2(yfl026w), rme1(ygr044c), mfalpha1 (ypl187w), mfα2(ygl089c), bar1(yil015w)*	*ste6, ste2, mfalpha1, mfalpha2, bar1*	mating associated
3	*dal7(yir031c), dal2(yir029w), dal4(yir028w), dur3(yhl016c), dur1(ybr208c)*	-	nitrogen metabolism

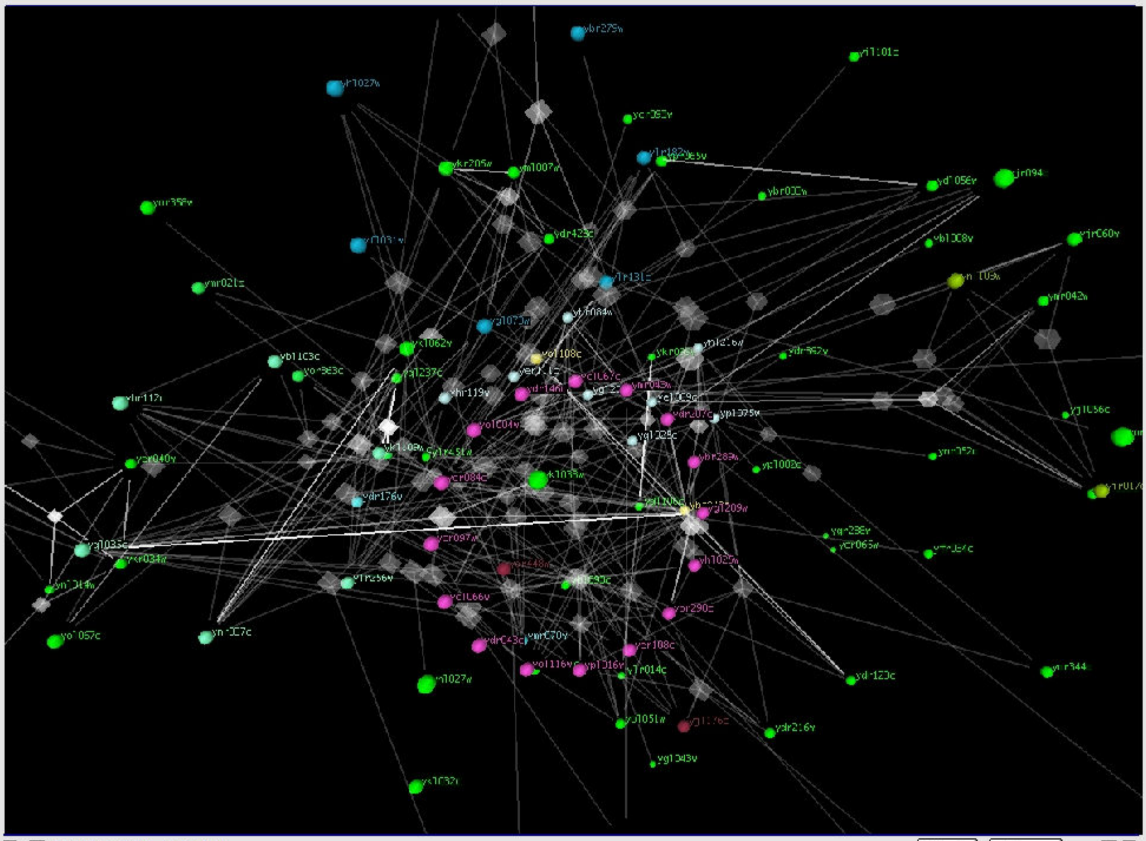
Table 4: Partitions with more than two vertices for different values of q in the complex of predecessors (core not shown). Informations about the functions of the genes have been obtained from the Saccharomyces Genome Database [ITCD+02].

Q-analysis: Some q-classes in the complex of successor

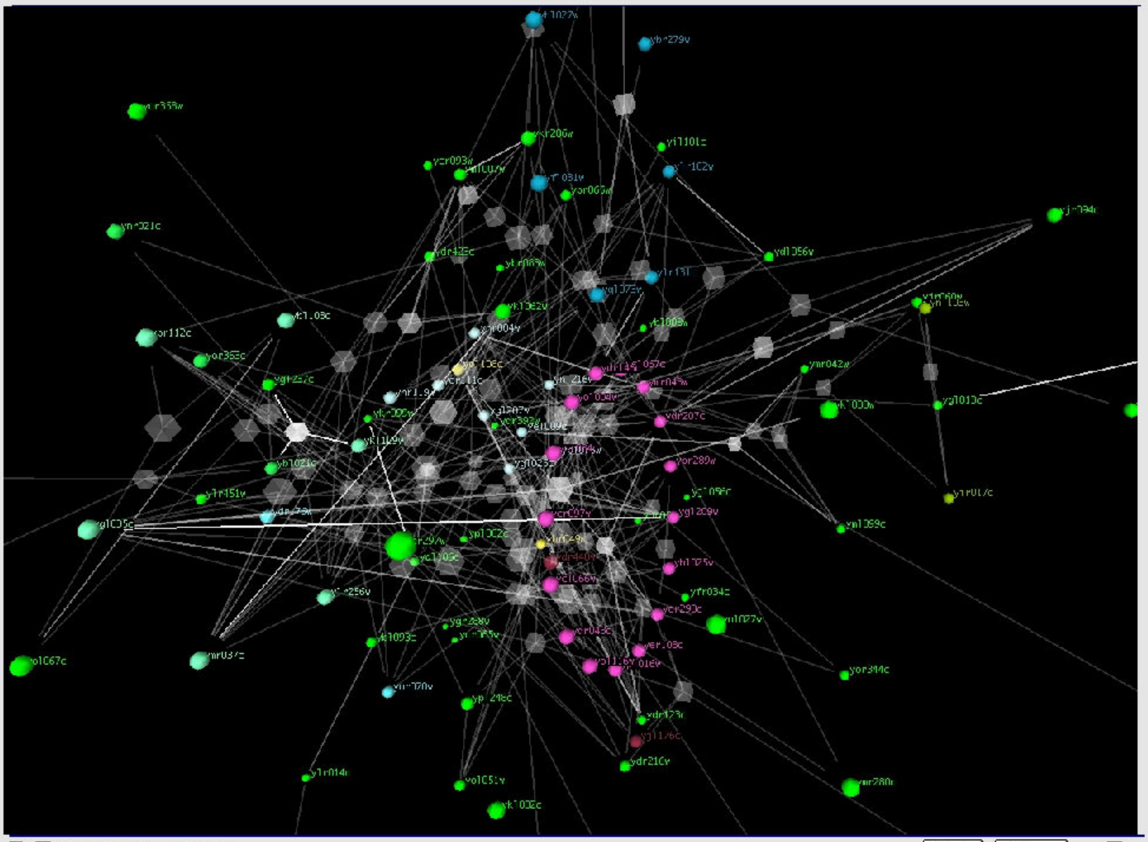
Partition	$q = 3$	$q = 4$	$q = 5$	Function
1	*swi1(ypl016w), snf2(yor290c), ino4(yol108c), swi3(yjl176c), snf6(yhl025w), ino2(ydr123c), snf5(ybr289w)*	*swi1, snf2, ino4, swi3, ino2*	*swi1, snf2, ino4, swi3, ino2*	chromatin remodeling, regulation of inositol/choline responsive genes
2	*swi6(ylr182w), swi4(yer111c), mbp1(ydl056w)*	*swi6, mbp1*	*swi6, mbp1*	transition from G1 to S phase in the cellcycle
3	*msn2(ymr037c), yap1(yml007w), ykr206w, msn4(ykl062w), hsf1(ygl073w), cad1(ydr423c)*	*msn2, yap1, ykr206w, msn4, cad1*	*YAP1,ykr206w, CAD1*	stress response
4	*dal82(ynl314w), dal80(ykr034w), gzf3(yjl110c), dal81(yir023w), gln3(yer040w)*	*dal82, dal80, dal81, gln3*	*dal82, dal80, dal81, gln3*	nitrogen metabolism
5	*bas1(ykr099w), gcn4(yel009c), pho2(ydl106c)*	-	-	regulation of amino acid metabolism and phosphate metabolism
6	*hap4(ykl109w), hap2(ygl237c), hap3(ybl021c)*	*hap4, hap2, hap3*	*hap4, hap2, hap3*	global regulation of respiratory gene expression

Table 5: Partitions with more than two vertices for different values of q in the complex of successors (core not shown). Informations about the functions of the genes have been obtained from the Saccharomyces Genome Database [ITCD⁺02].

Q-analysis: Visualization through clustering by Rips complex



Q-analysis: Visualization through clustering by Rips complex



Programming the interaction structure for simulation

- **A data structure is a space: *topological collections***
  a *chain* = a complex where values are associated to cells

- **A control structure is the specification of the interacting parts**
  an *interacting part* is a *subspace*, so
  a *path* in the space
  a *sub-chain*

- **A *transformation* replaces somes parts by other parts**
  abstract rewriting in topological space

- **A rule-based language: MGS**
  fully embedded in a "true" functional language
  transformation are case-based function definition
  topological collection extend "algebraic data type"
  statically or dynamically typed: $[\alpha]\theta$
  interpretation or compilation

MGS

un Modèle Général de Systèmes de transformations

+ a collection is a set of values with some *topology* that defines the notions of *subcollection* and *boundary of a subcollection*

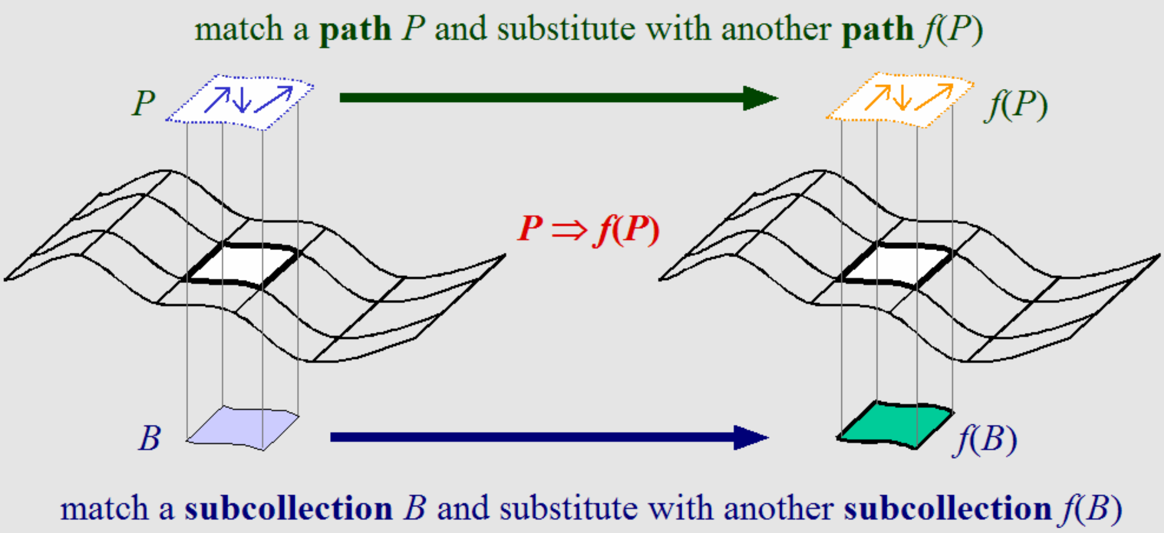
+ a *transformation* applies locally by replacing a subcollection by another which has *compatible boundary*

+ a transformation is specified by a set of *rules*

+ a computation consists to *iterate* a *transformation* on a *collection*

and taking into account the necessary features of a usable programming langage

+ managing the iterations and mastering the rules applications
+ rich familly of base collections and topologies
+ expressive language for the specification of the subcollection to be replaced



A path induces a subcollection.
A path is more easy to handle: it is a sequence

x neighboor of *y*

satisfying *x*≤*y*

is replaced by

```
T =
{
  (x,y / x ≤ y) => f(x,y)
}
```

pattern used to filter the subcollection to be replaced

a computed subcollection

a subcollection s *of less than 10 elements whose sum is greater to 5*

```
x+ as s / (Cardinal(s) < 10) & (Fold[+](s) >= 5)
```


How to specify a part in the neighborhood graph (complex) of a data structure

path patterns

Matching One Element

`_`	- matching an anonymous element
x	- matching an element and binding its value to a name **x**
x/C	- matching an element satisfying some condition **C**

Path Pattern (PP)

x	- an element
x,PP	- an element followed in any direction by a PP
x|g>PP	- an element followed in some direction **g** by a PP

Repetition (*, +)

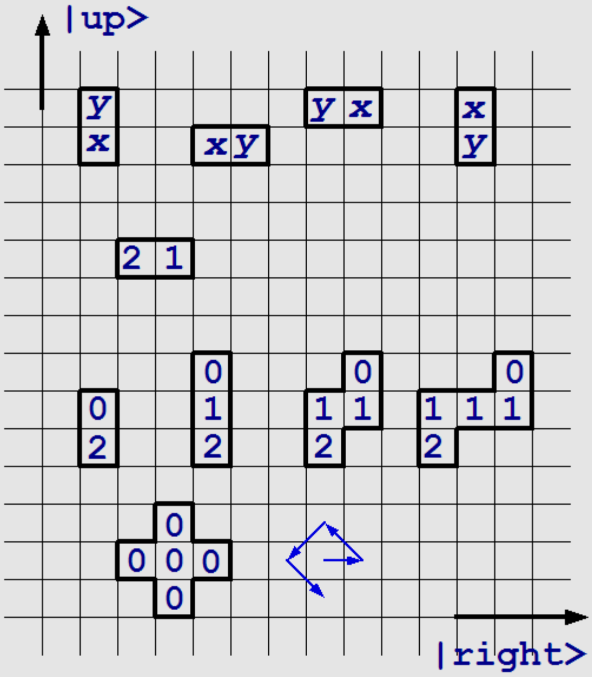
x,* or **x***	- repetition of an element following any direction
x|g>*	- repetition of an element following a direction **g**

x,y

x |right> **y** /**x>y**

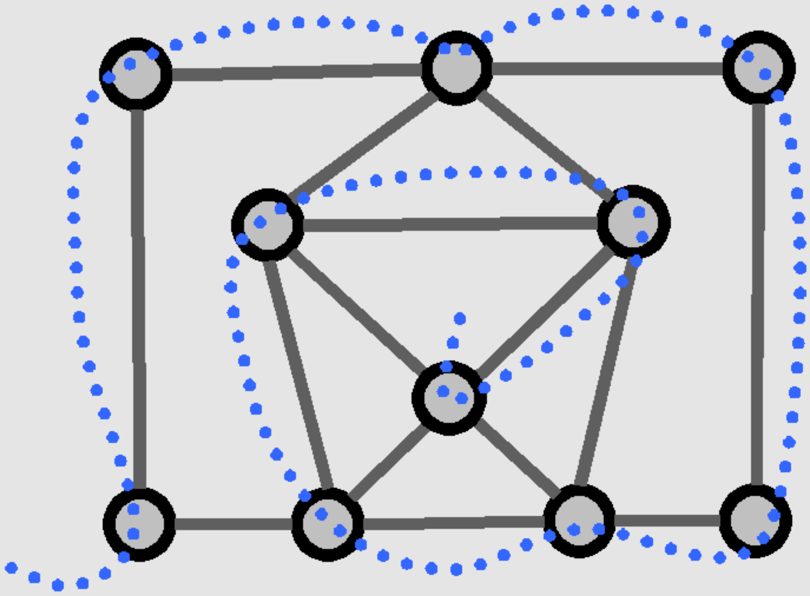
x |up>(1|right>*)|up> **y** /**x>y**

0|right>
1|up-right>
2|-up-right>
3|right-up>
4



```
H = {x* / size(x*)= n => Print(x*)}
```

a path matched by a pattern has no repetition

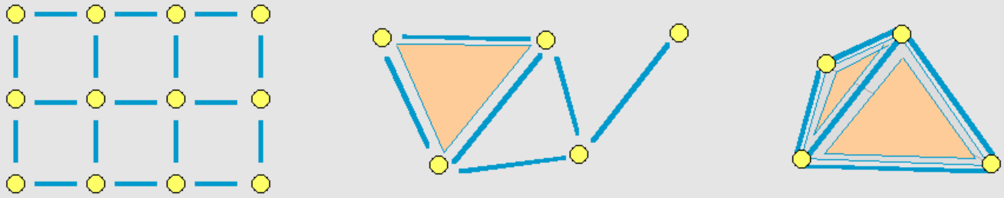


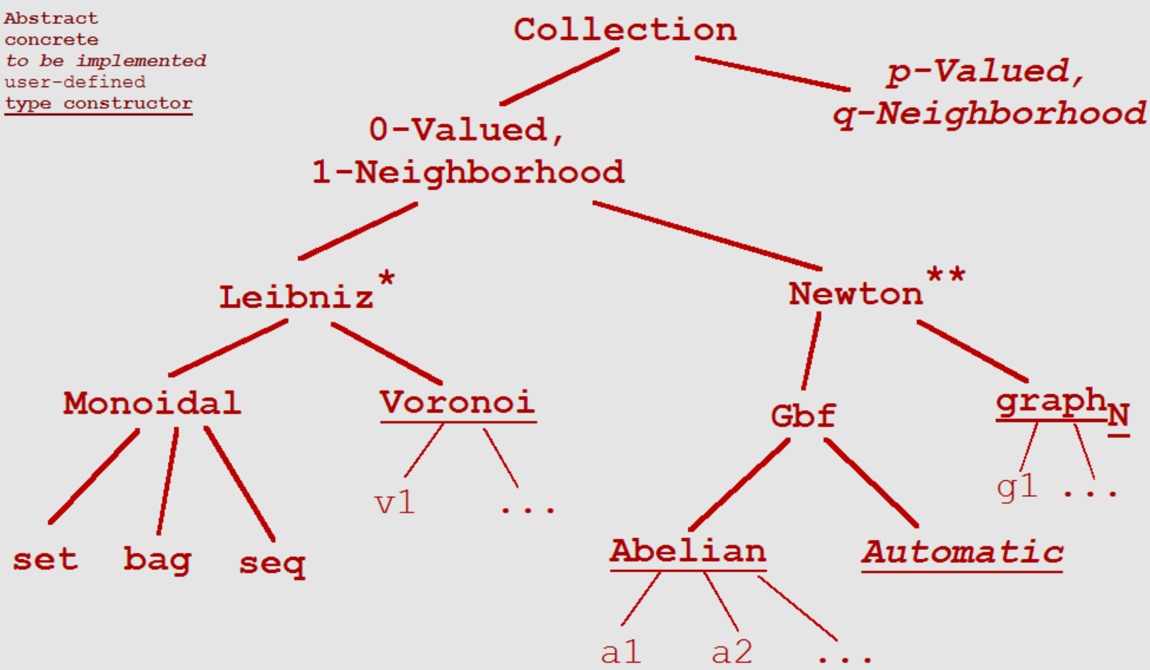
By changing the collection type (really the topology), we obtain several computational models

- scalars et finite products of scalars ≅ Stream (Otto, dataflow)
- set
- multiset ≅ CHAM
- multiset + nesting ≅ P-system
- sequence ≅ L-systems, string rewriting, DNA computing
- array ≅ Cellular Automata
- lambda-term ????

to handle arbitrary abstract topology in full generality

- chain complex





*Leibniz: **x => <undef>** means delete

Newton: **x => <undef> means undefined value at position **x**

Control of the rules application

- an expressive pattern-matching language
- powerful control mechanisms to manage the application of transformations and their iterations

transformation = `{ rules }`

priority :
```
x:P, y ={priority = 2}=> x+y
x, y:Q ={priority = 1}=> x*y
```
parallel rule :
```
{x=u} +=> {x=u+1}
{y=v} +=> {y=2*v}
```
side effects :
```
T[a=0] = { x => (a:=a+1; x+a) }
```
guards :
```
x ={on mode ~= apoptosis}=> (a:=a+1;x+1)
```
application :
```
T[n], T['fixpoint], T['fixrule], T[a=5],
T['asynchrone], T['stochastic]
```
etc.

Some simple examples (algorithmics)

Application to "classical" algorithmic

- **cf. Gamma algorithmic:**
  prime numbers, sorting, optimization problems (from knapsack to maximum segment sum), convex hull, etc.
- **Graph algorithms:**
  shortest path, hamiltonian path, maximal flow, etc.
- **List algorithms:** fold, map, etc.

"classical" discipline

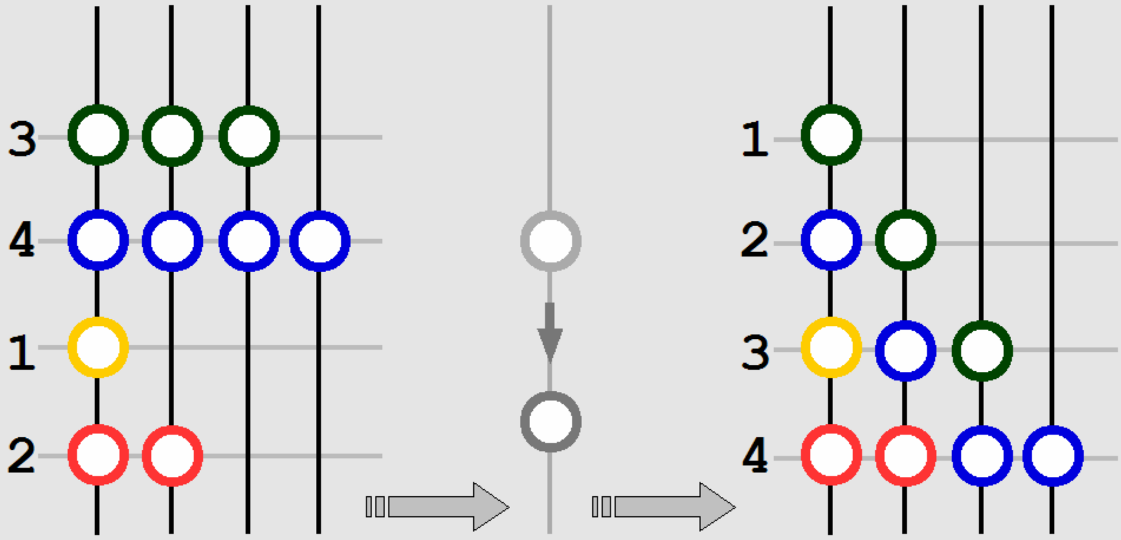
map = trans[f] { x => f(x) } : $\alpha \to \beta \to [\alpha]\theta \to [\beta]\theta$

trans { x => x+1 } : $[\texttt{int}]\theta \to [\texttt{int}]\theta$

trans { x:int => x+1 } : $[\alpha]\theta \to [\alpha]\theta$

trans { x / leftQ(x) => x } : $[\alpha]\texttt{seq} \to [\alpha]\texttt{seq}$

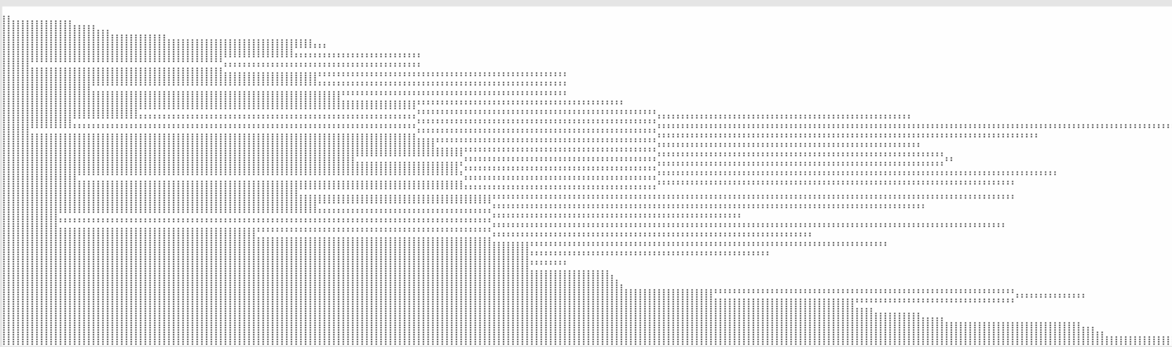
Examples (algorithmics): Bead sorting



Arulanandham, J. J.; Calude, C. S.; Dinneen, M. J. *Bead-Sort: A Natural Sorting Algorithm.* UMC'02, Kobe, Japan, October 15-19, 2002.

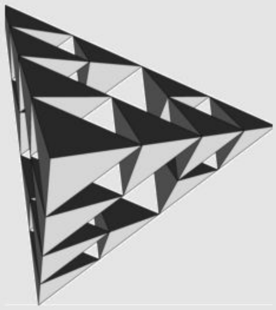
Examples (algorithmics): Bead sorting

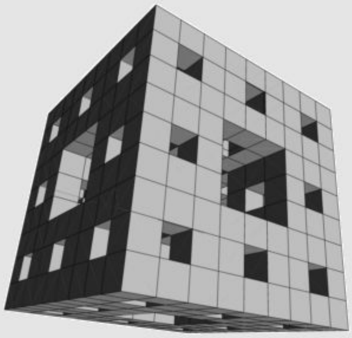
```
// Defining a 2D neighborhood
gbf abacus = < rods, levels >;;

// The transformation that lets "fall" the beads
trans do_sort = {
    "|" |rods> " "  =>  " ", "|"
};;

// Sorting is to let fall the beads until fixpoint
fun sort(liste) = do_sort[*](prepare(liste)) ;;
```



ST by carving in MGS

- requires *patches* and more complex patterns
- the result (2 steps of the sponge in 75 secs; 4 steps of the sierpinski in 33 secs)





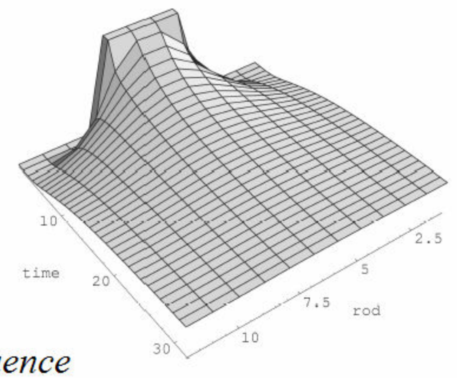
Some simple examples (simulations)

Application to the simulation

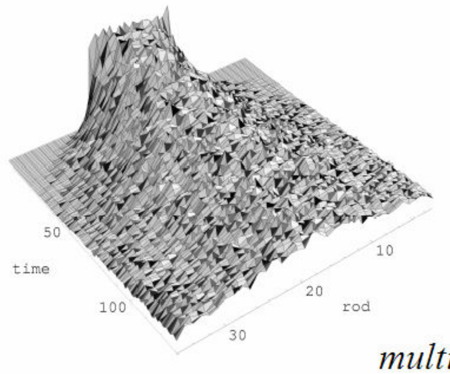
WHY: many mathematical, physical, and biological systems are based on a notion of state that associates data to each point of an abstract or a physical space.

MGS is designed to specify the dynamical construction of these spaces and to handle the associated data.

physic => dynamical systems
biology => dynamical systems **with a dynamical structure**



sequence



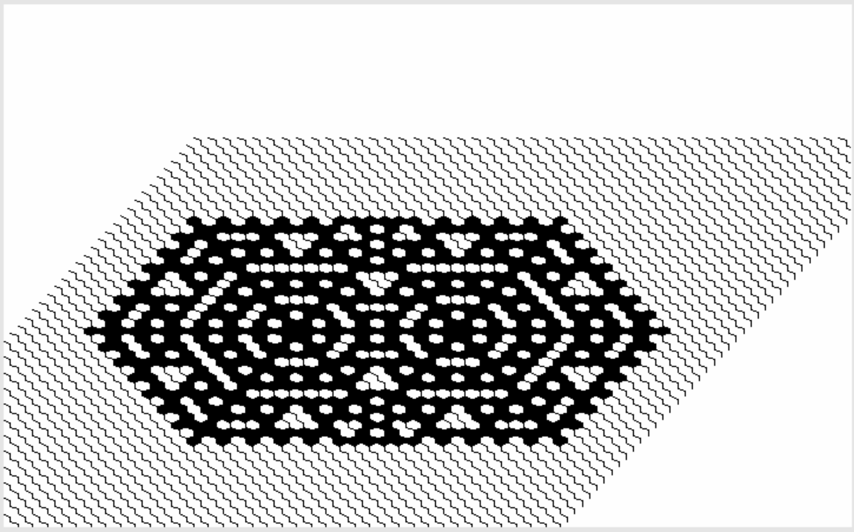
multiset

```
trans Heat = {
    y => let x = left(y)
         and z = right(y)
         in
          h*y + (1-h)/2 (x+z);
    y / not(leftQ(y)) => ...
    y / not(rightQ(y)) => ...
}
```

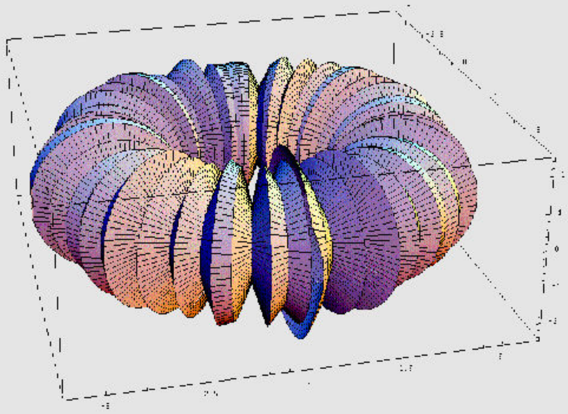
```
trans Heat = {
    y / y != 0 and y != max
    => random(y-1, y, y+1);

    0 => random(0, 1);

    max => random(max-1, max);
}
```

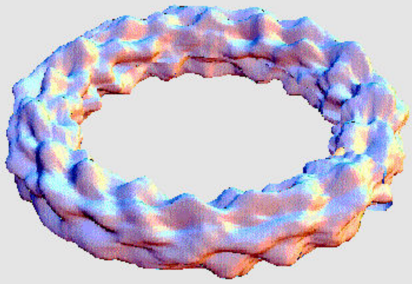

Snow flake on an hexagonal lattice

```
gbf hexa_grid = <a,b,c; a+b=c>;;

trans T = (0 as x / (neighborsfold((\a,b.a+b), 0, x)==1)=> 1);;
```



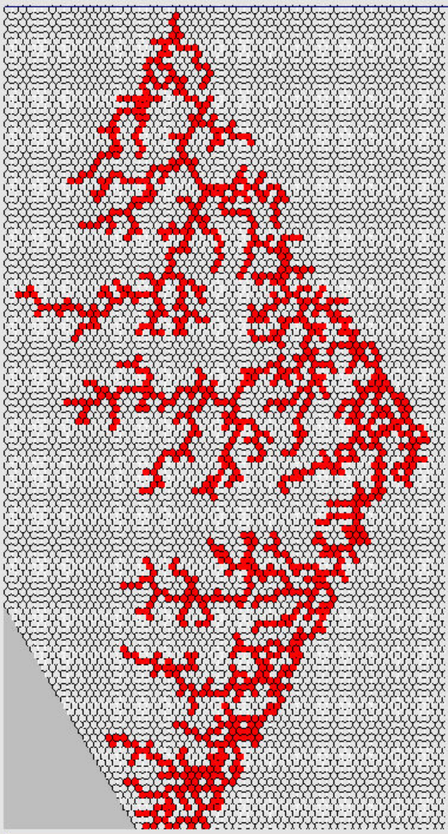
Examples (DS)2: Turing diffusion-reaction





```
gbf ring = <left, right; left+right = 0, 40*left = 0>

trans Turing[rsp = 1.0/16.0, diff1 = 0.25, diff2 = 0.0625] =
{
    x => { a = x.a + da(x.a, x.b, left(x).a, right(x).a),
           b = Max(0.0, x.b + db(x.a, x.b, x.beta, left(x).b, right(x).b))  };
};;
```


Examples (DS)2: Diffusion limited aggregation

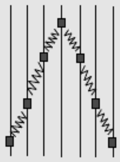


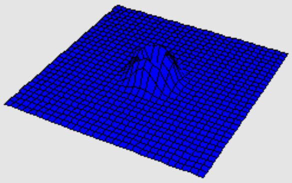
```
trans DLA = {
  'free,'fixed => 'fixed,'fixed
  'free,'empty => 'empty,'fixed
}
```

Examples (DS)2: Amorphous wave equation

Amorphous : local connectivity, no global communication, unreliable substrate

Model described by two variables : *amplitude* and *momentum*



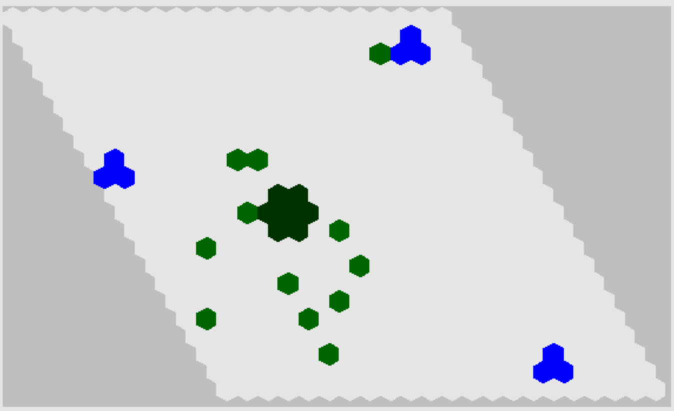


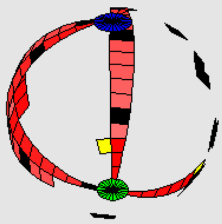
Polytypism: same transformation applied to different spaces

(read: same function on different data type)

ErikRauch - *Discrete, Amorphous Physical Models* – International Journal of Theoretical Physics - 2001.

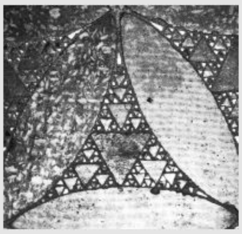
Examples (DS)2: Ants foraging





Shortest path by ants foraging

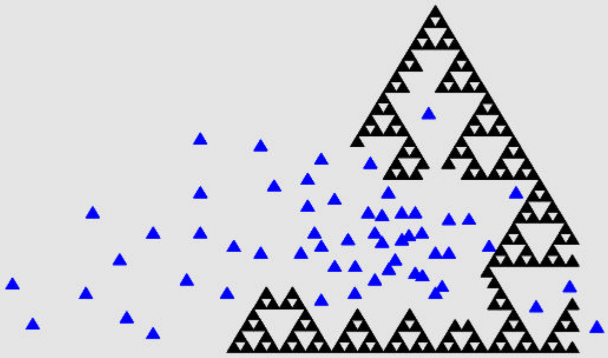
Sierpinski triangle (ST)



- Description
  - Fractal
  - Appearing from the 13th century (Anagni cathedral in Italy)
  - Waclaw Sierpienski in 1915

- In MGS by

  Accretive growth
  - GBF collection
  - Path transformation



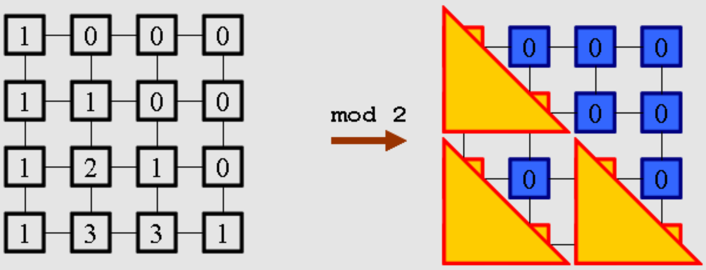
...

ST by accretion

- Self-assembly by accretive growth
  - Basic elements aggregating into a shape
  - Material is added in each growth stage
  - Growing process takes place on the boundaries (not *intercalary growth*)

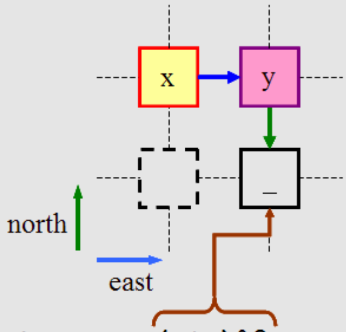
- ST by accretion

  produced by Pascal's triangle mod 2: $P(i, j) = P(i\text{-}1, j\text{-}1) + P(i\text{-}1, j)$



ST by accretion in MGS

- $P_{i+1,j+1} = (P_{i,j} + P_{i+1,j}) \% 2$



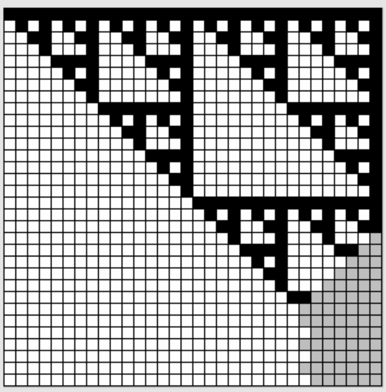
- MGS code

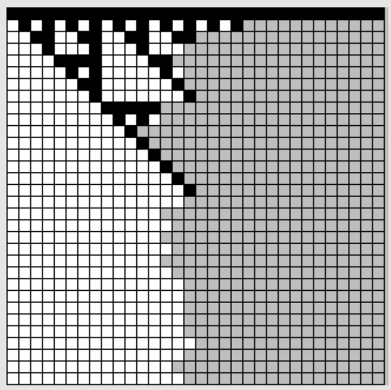
```
trans ST = {
  x |east> y |-north> <undef> => x, y, (x+y)%2
};;
```

- or following [Rothemund04] DNA's xor automata

```
trans Tiling = {
  (`T00|`T11) as x |east> (`T01|`T10) as y |-north> <undef>
      => x, y, `T01;
  (`T00|`T11) as x |east> (`T00|`T11) as y |-north> <undef>
      => x, y, `T00;
 …} ;;
```


ST by accretion in MGS





parallel maximal rule application strategy

stochastic rule application strategy

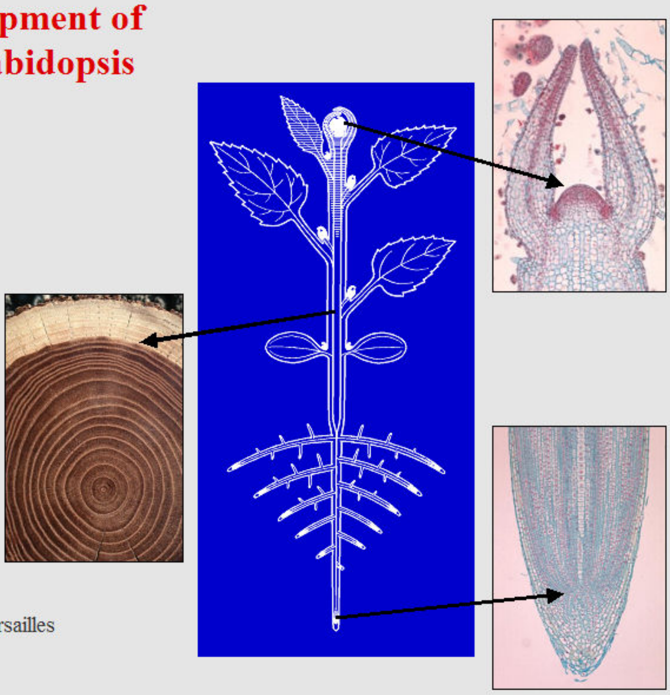
Modelisation of the development of the apical meristem of Arabidopsis Thaliana

Pierre Barbier de Reuille[1]

Mikaël Lucas[2]

Jan Traas[3]

Christophe Godin[4]

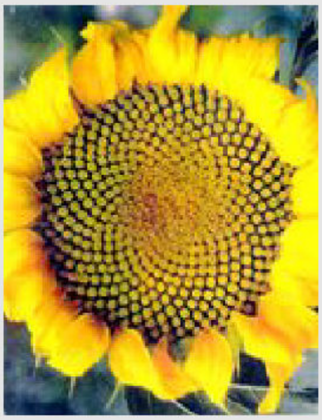


[1] INRA, UMR AMAP Montpellier

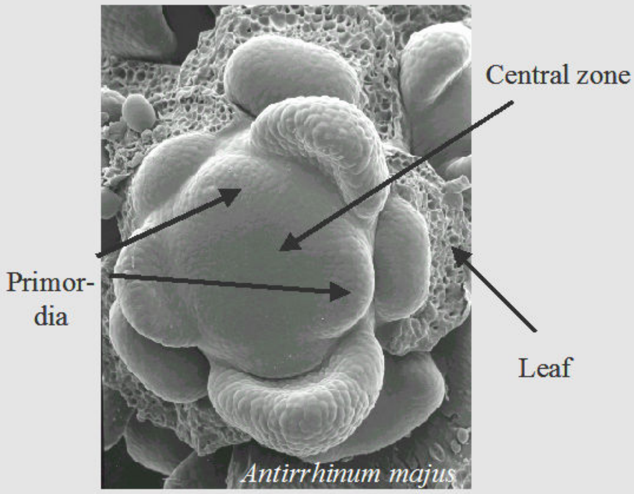
[2] ENS, Lyon

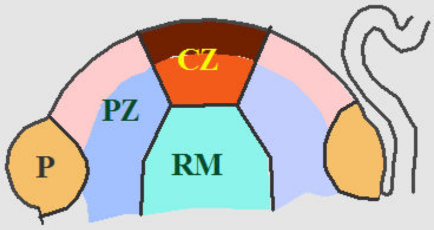
[3] INRA, Laboratoire de biologie cellulaire de Versailles

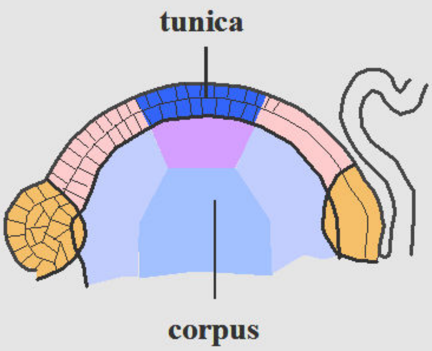
[4] INRIA, UMR AMAP Montpellier



Modeling the meristem's growth: Primordia

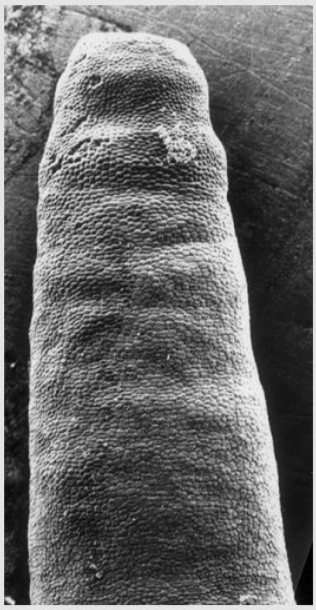






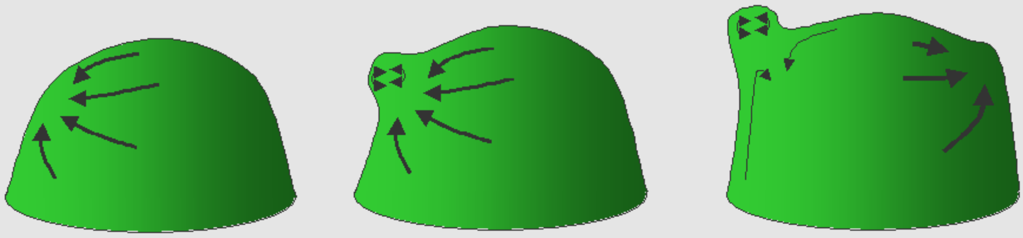
Modeling the meristem's growth: role of PIN 1

The perturbation of the auxin transport is correlated with the perturbation of the formation of the organs in the PIN1 mutant

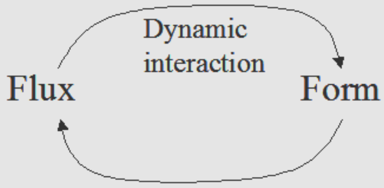


normal plant **PIN 1**

Modeling the meristem's growth: A dynamical system with a dynamic structure



Flux… changes Form… which changes flux…



Modeling the meristem's growth: Modelization choice

- Representation of only the L1 layer

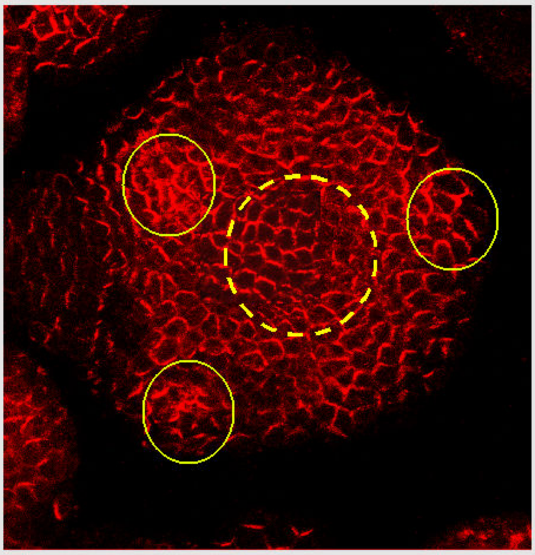
- 2D model of the transport

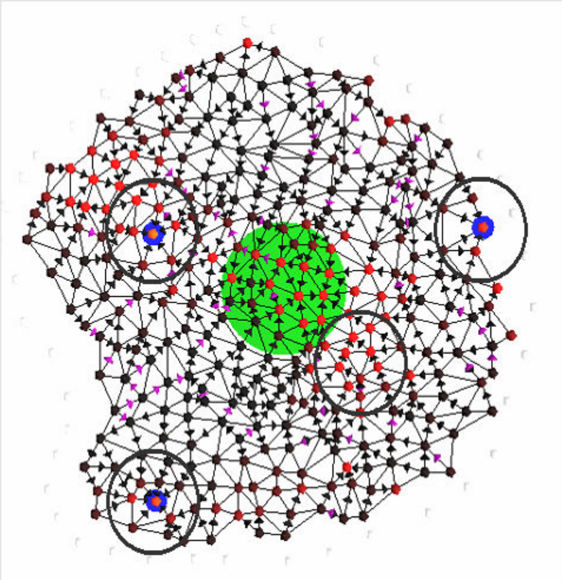
- Discrete modelization of the cells (Voronoi tessalation)

  growth, division, exchanges between cells

state of the cell	=> ability to divide, strength of the spring, concentration in the inhibitor and auxin
growth	=> increment of the strength, function of the inhibitor concentration
movement	=> only due to the growth
division	=> when the size is above a given threshold
cellular interaction	=> transport and diffusion of auxin, *passive diffusion of the inhibitor*

Modeling the meristem's growth: Result of the simulation





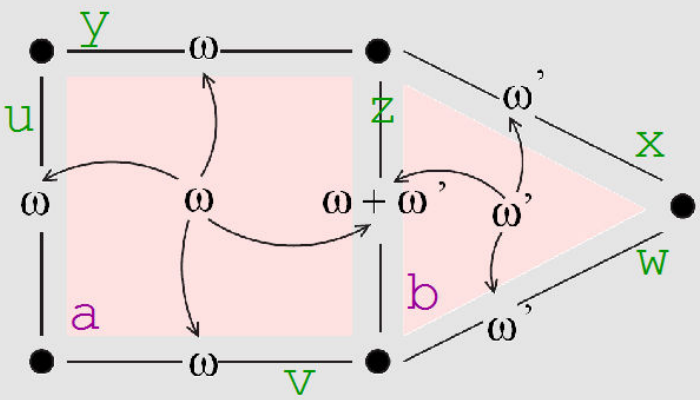
blue circles : high concentration of auxin

High concentration of auxin in the CZ (and remains even by changing the parameters)

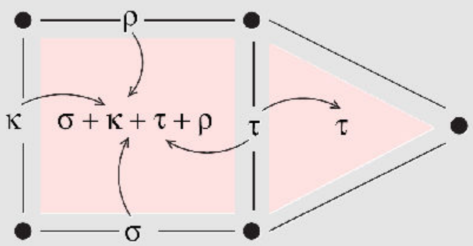
From global to local: rule and differential operators

Boundary and coboundary operators as transport operators

$$\partial(\omega.a + \omega'.b) = \omega(u+v+z+y) + \omega'(z+w+x)$$



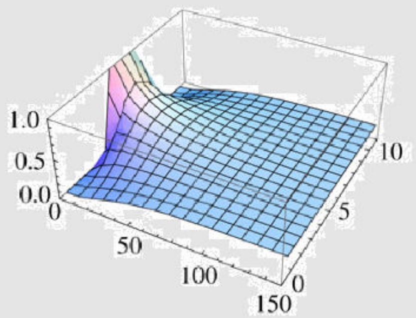
Coboundary d



From global to local: Boundary, Coboundary, Laplacian... are MGS transformation

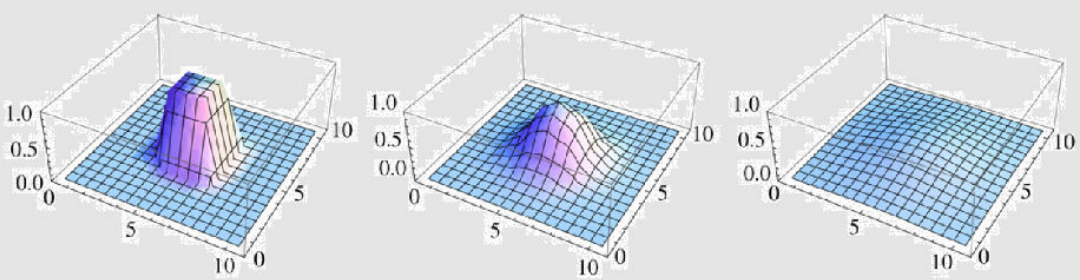
```
trans boundary[ addition = \(S,e,acc).({| e@oneof(S) |}, acc),
                zero     = {||}
              ] =
{  x => cofacefold(\(y,acc).(orient(^y,^x,y) +_val acc), <undef>, x)  }


trans coboundary[ addition = \(S,e,acc).({|e@oneof(S)|}, acc),
                  zero     = {||}
                ] =
{  x => facefold(\(y,acc).(orient(^y,^x,y) +_val acc), <undef>, x)  }

fun grad(V) = coboundary<1>(V)
fun rot(c) = coboundary<2>(c)
fun div(c) = coboundary<3>(c)

fun coderivative[zero=<undef>, addtion=\(S,e,acc).(e +_val acc)](T) =
  \c.T[zero=zero,addition=addition](coboundary(c))
;;

fun laplacian[zero=<undef>, addtion=\(S,e,acc).(e +_val acc)](T) =
  (derivative(coderivative(T)))(c), (coderivative(derivative(T))(c))
;;
```


From global to local: Examples



Simulations of heat diffusion in 1D
the picture on the left shows the evolution of temperature along a 1D rod divided into 11 blocks during 150 units of time
and 2D
the three pictures on the right present respectively the temperature distribution on a 11×11 square grid at the initial state, after 50 steps and after 300 steps.

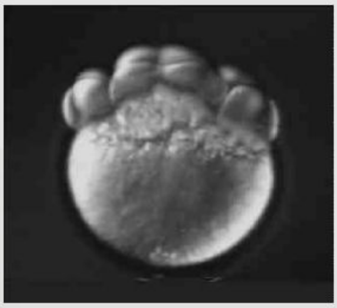
J.-L. Giavitto, A. Spicher / Physica D 237 (2008) 1302–1314

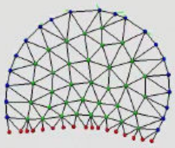


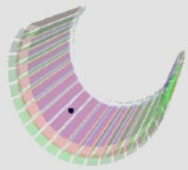
Achievements, futur work and a manifesto

- The MGS prototype
  http://mgs.ibisc.univ-evry.fr
- MGS handles:
  set, multiset, sequence,
  GBF,
  arbitrary graphs,
  Voronoi neighborhood
  GMAP
  abstract cell complex
  and the arbitrary ***nesting*** of such structures
- The actual Path Pattern language is pretty sophisticated:
  `(x/x>5)+ as X / Fold(+,0, X) < 25 -> ...`
- There is a Patch Pattern language in progress

- **typing** of topological collection and their transformation
- **compilation** of path matching (TOM, others...)
- extension of the pattern matching facilities (failure, eager/lazy matching strategy, pattern sharing, ...)
- complexity of patterns and optimization
- algebraic structure of the set of paths
- implementation and validation
- **Autonomic, global-to-local systems specification and programming** (application to synthetic biology)







A topological Manifesto

The logical approach is fertil :

Logic: computation = deduction

other paradigmes are fruitful too:

Topology: computation = moving in a space

Try to perceive space (and time) in a program

purposes:

teaching, technical and heuristical

michel@ibisc.univ-evry.fr
giavitto@ibisc.univ-evry.fr