Algorithmic Examples

Multiset and the Chemical Model

The Chemical model

- Gamma (Banâtre & Le Metayer, 1986):
  - Data: “floating” molecules in the solution
  - Computation: chemical reactions between the molecules
  - no over specification of control structure (non determinism, parallelism)
  - no over specification of data structure (multisets as blackboard)

- Metaphor / Foundations

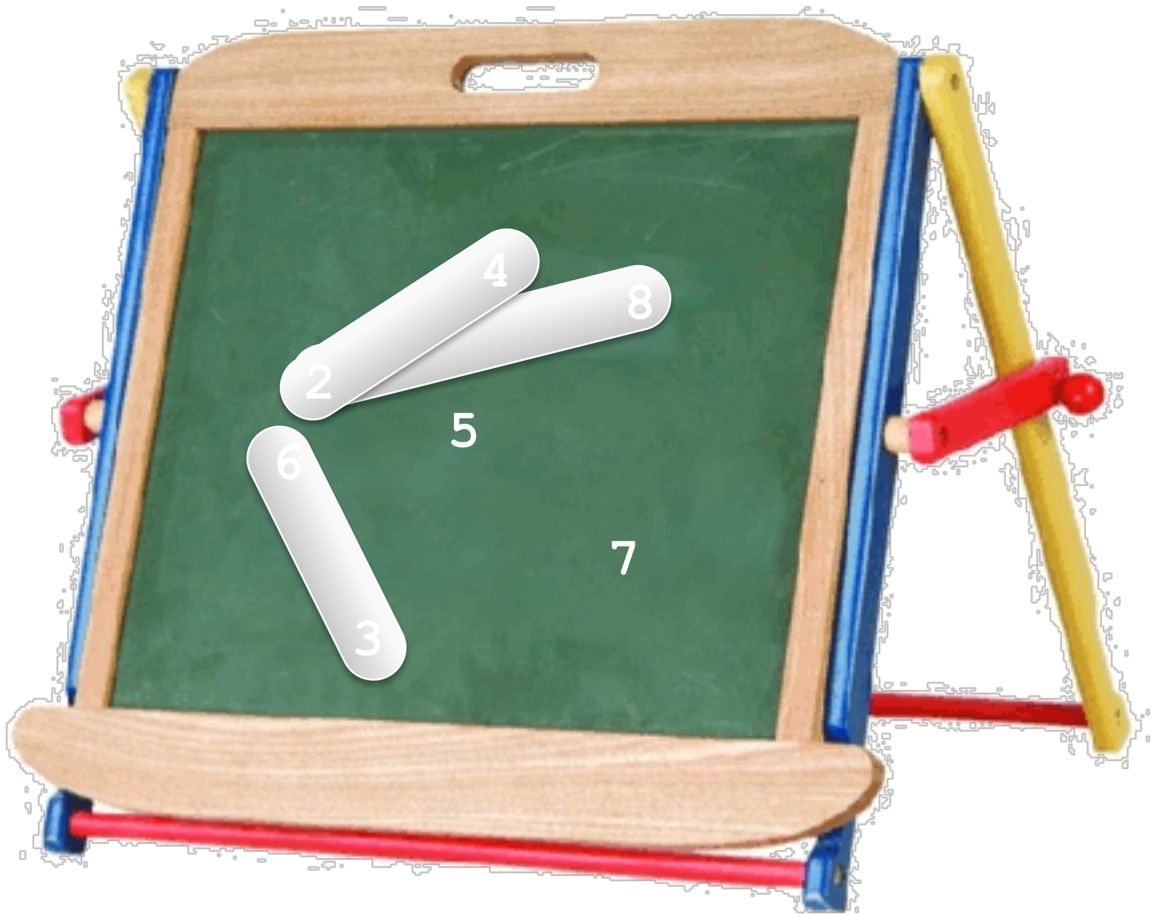
Metaphor	Foundations
Solution	Multisets
Reactions	Rewriting rules
Bownian motion	Associativity/Commutativity

- Implicit parallelism and autonomy of reactions until inertia

Computing the primes

replace *x*, *y* **by** *x* **if** *x* divide *y*

steady state



Pro and Cons

- Very high-level languages

- Between specification and programs
  - not specifications
    (*e.g.,* several algorithms are expressible for the same task)
  - not programs
    (typically not the same complexity)

- Relevant for the programming of large autonomic and distributed systems
  1. The multiset data structure and rewriting suitably represent the orderless interactions (reactions) between elements that occur in large parallel or open systems
  2. Autonomic properties (e.g. self-healing, self-protection, self-optimization, etc.) are naturally expressed as reaction rules. The corresponding behavior can be seen as the corrective action corresponding a perturbation.

Pro and Cons

The high-level nature of chemical programming entails also drawbacks

1. multisets are weak data structure
   - difficult to express data structure
   - difficult to express selection and control of rules
   - difficult to represent the distribution
     - neighborhood relationships represent physical constraints (spatial distribution, localization of the resources)
     - multiset = ether

2. AC rewriting can be inefficient
   - the selection of elements
   - the ordering of reactions
   - the termination

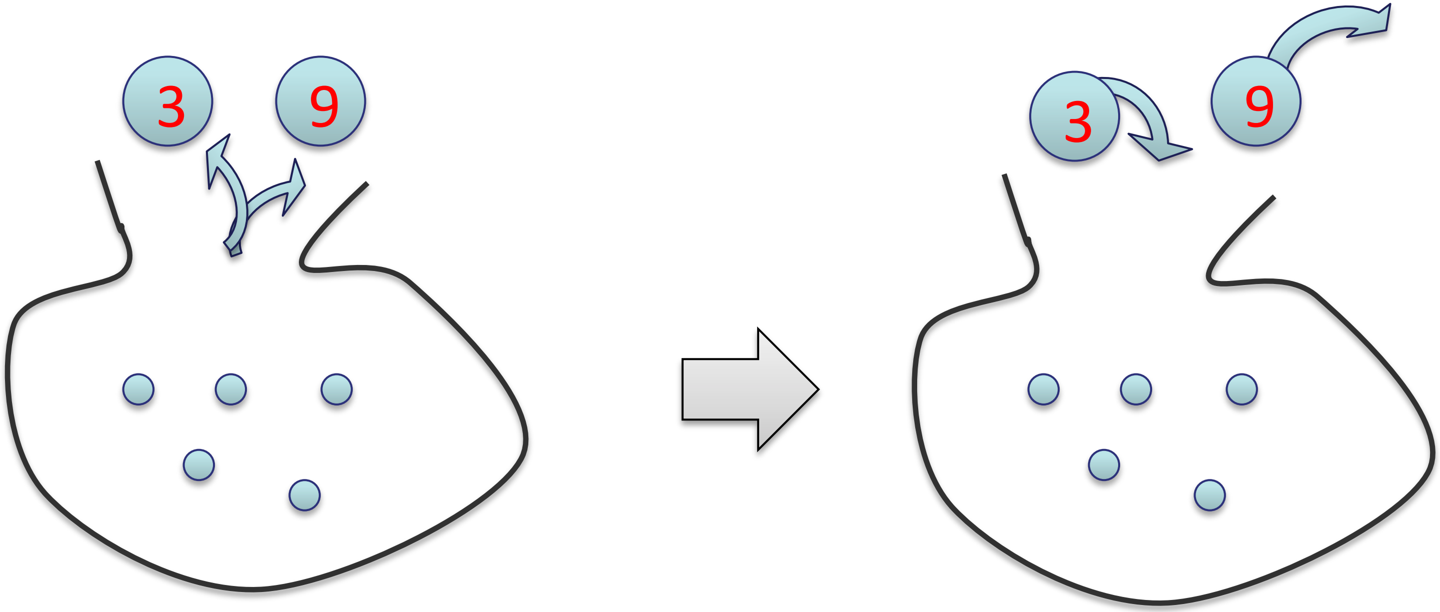
Structured Gamma

MGS

HOCL

Semi Automatic Refinement

Eratosthene's Sieve



$$\texttt{trans}\ \mathit{Generate} = \{x, \mathit{true}\} \Rightarrow x, \{x+1, \mathit{true}\};$$
$$\texttt{trans}\ \mathit{Succed} = \{x, \mathit{true}\} \Rightarrow x;$$
$$\texttt{trans}\ \mathit{Eliminate} = (x, y\ /\ y \bmod x = 0) \Rightarrow x;$$

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

Eratosthene’s Sieve

```
trans X = { x, y / (x%y == 0)  =>  y }
```

applied on the bag

```
2, 3, 4, 5, 6, …, n
```

(in a bag, any elements are neighbor)

At fixpoint, there is no *x,y* such that *y* divides *x*.
That is: the numbers in the multiset are relatively primes.
And because we started from all number between 2 and *n*,
we have the **primes below *n***.

Sequence

Path transformation

Example: "bubble sort" of a sequence

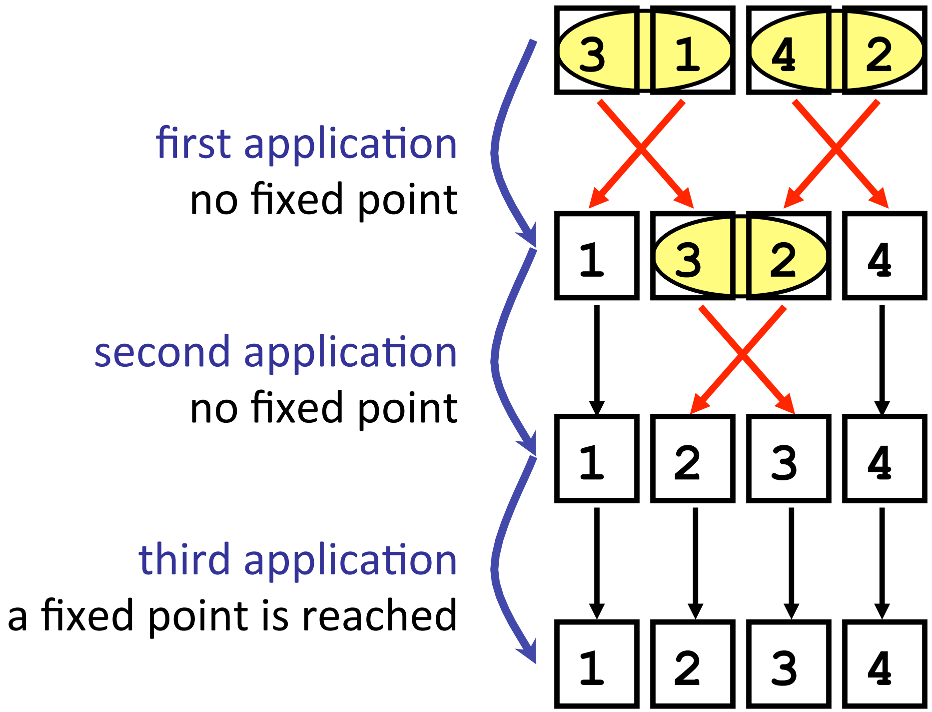
```
trans bubble_sort = {   x , y / (x > y) => y , x   };;
```

Select x **neighbor of** y **such that** (x > y)

A maximal parallel pattern matching

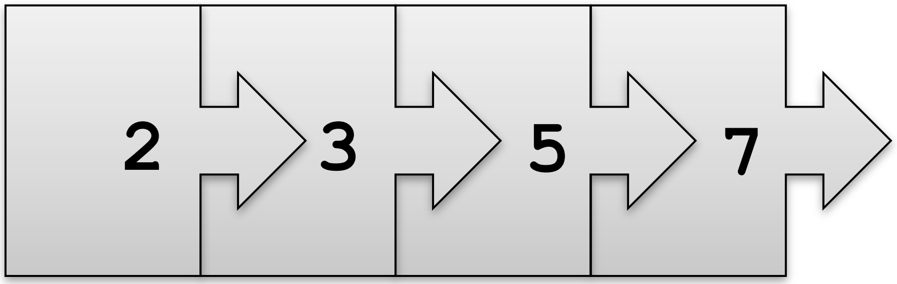
Example: "bubble sort" of a sequence

```
trans bubble_sort = {    x , y / (x > y) => y , x    };;
bubble_sort[`fixpoint]((3,1,4,2)) ;;
```

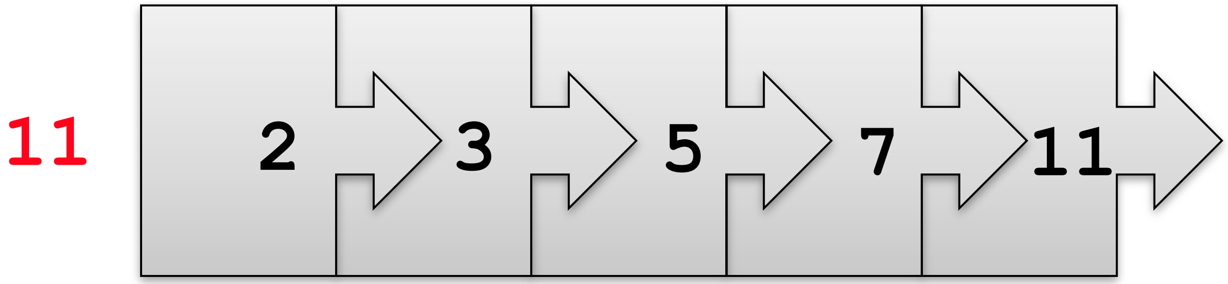


Eratosthene's Sieve

9



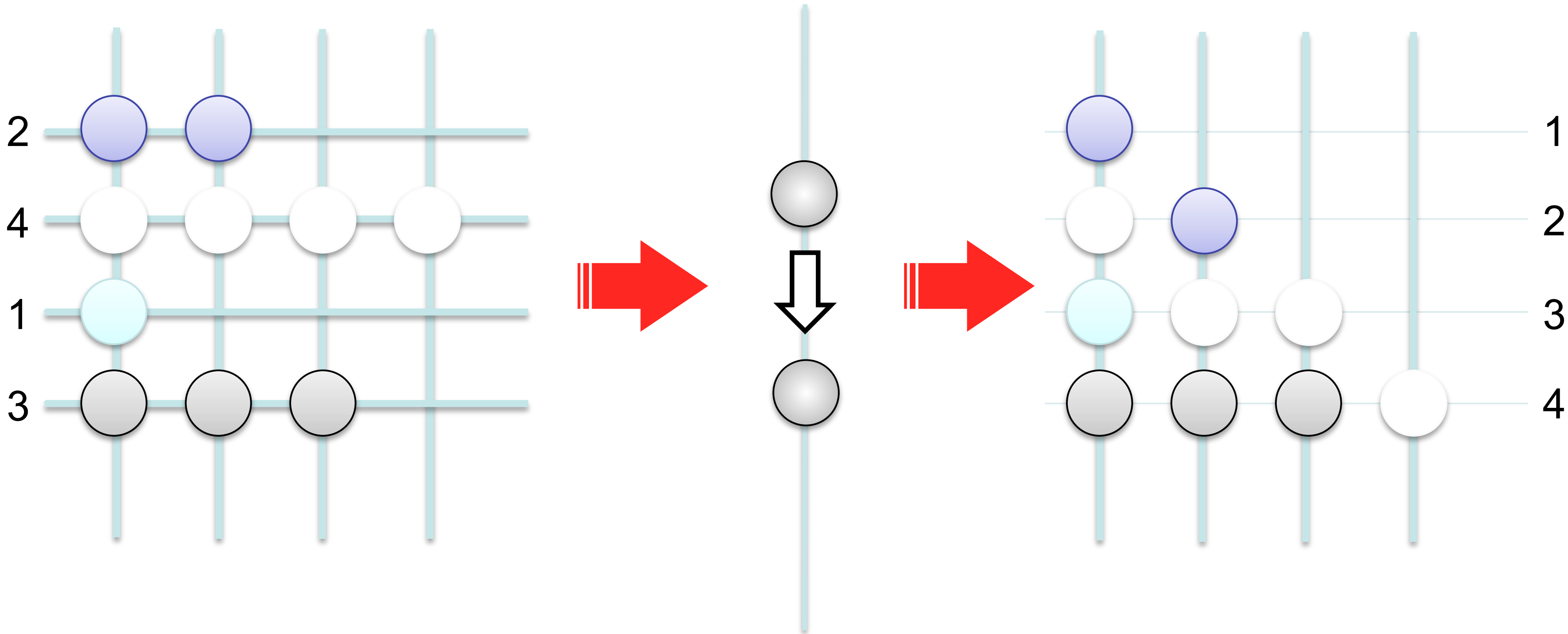
Eratosthene’s Sieve



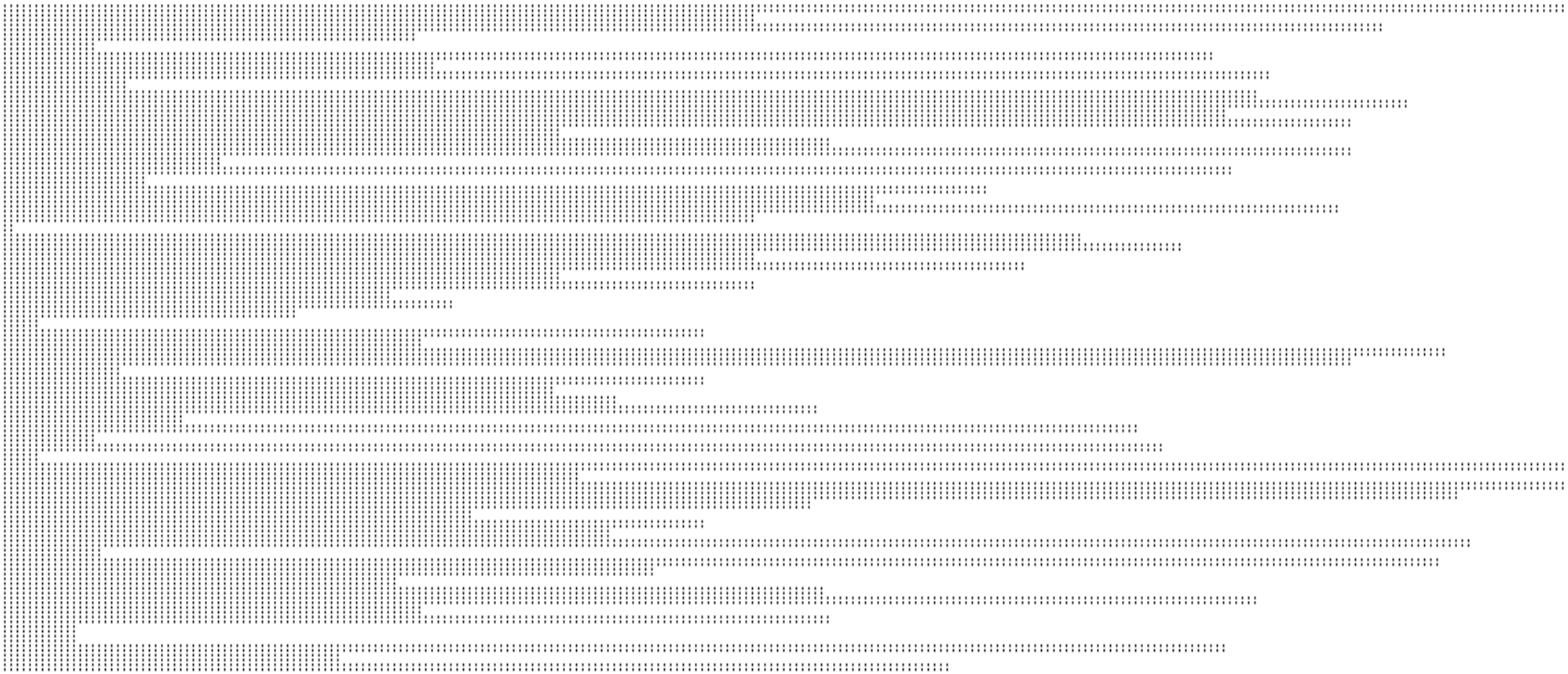
$$
\begin{array}{lll}
\texttt{trans}\ Eratos = \{ & & \\
\quad Genere1 & = & n : integer\ /\ \tilde{}\,\texttt{right}\ n\ \ \texttt{=>}\ \ n,\ \{prime = n\}; \\
\quad Genere2 & = & n : integer, \{prime\ \texttt{as}\ x, \tilde{}\,candidate, \tilde{}\,ok\} \\
 & & \texttt{=>}\ \ n+1,\ \{prime = x, candidate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

Array

Bead sort



Bead sort



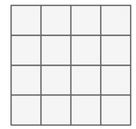
```
trans bead_sort = { • |south> <empty> => <empty>, • };;
```

Group based fields

From Arrays, Data Fields and GBF to Chain

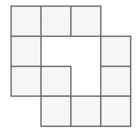
Array
(Total Function)

$$[0, d_1] \times \ldots \times [0, d_n] \longmapsto Val$$



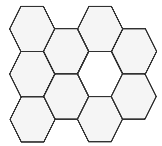
Data Field
(Partial Function)

$$Z^n \longrightarrow Val$$



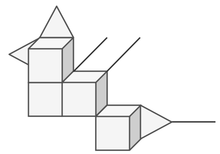
GBF (Group based Field)
(Partial Function)

$$Group \longrightarrow Val$$

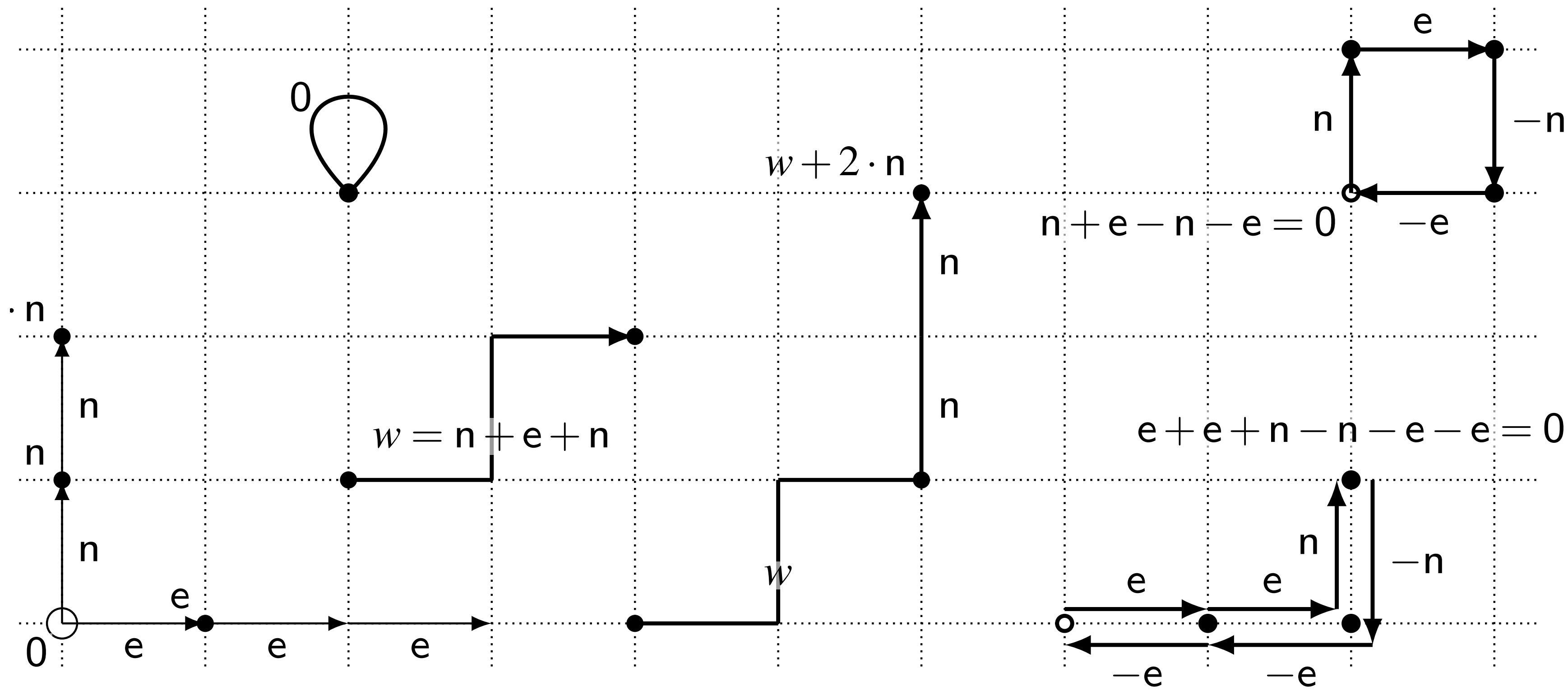


Chain
(Partial Function)

$$Cellular\ complex \longrightarrow Val$$



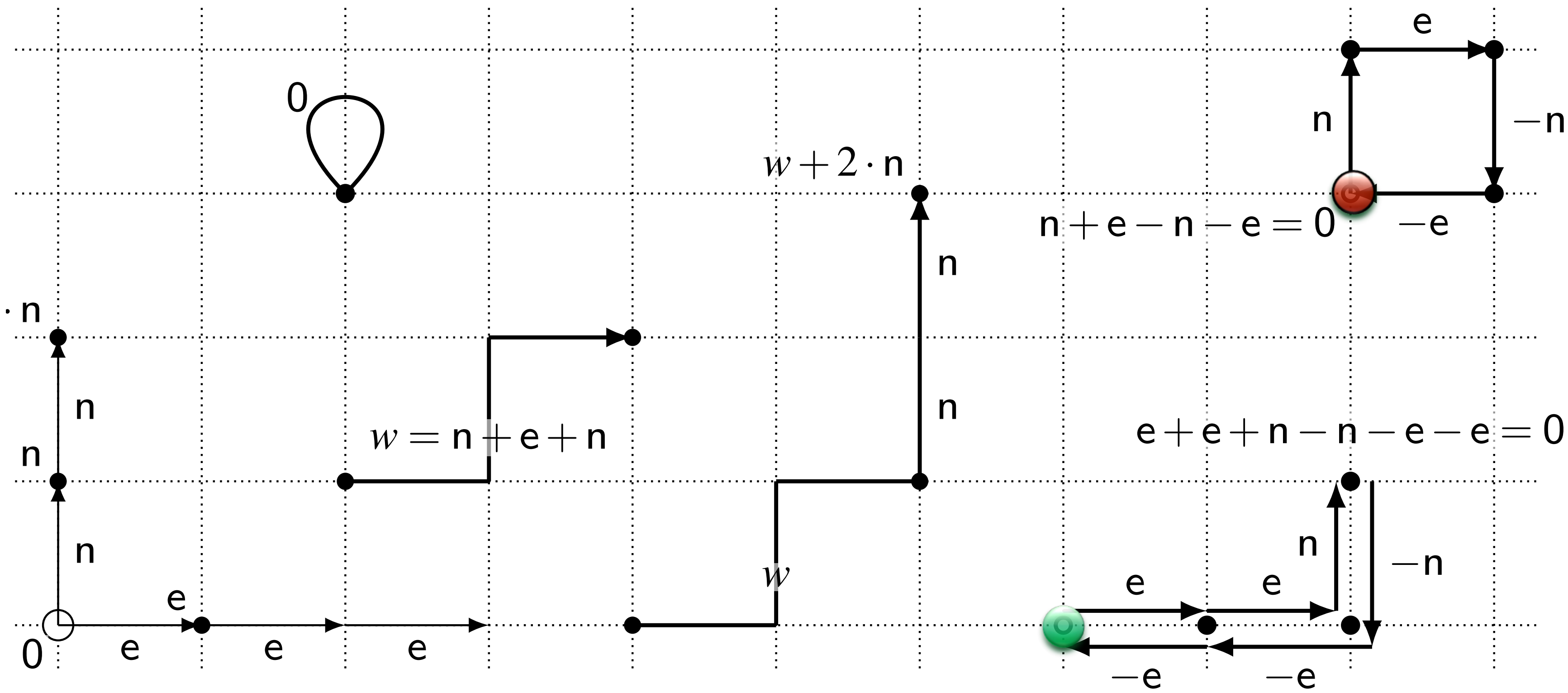
GBF



Cayley graph of a finite group presentation: < n, e; n+e=e+n >

- vertices are group element
- edges are generators *g* linking u and v iff u + *g* = v

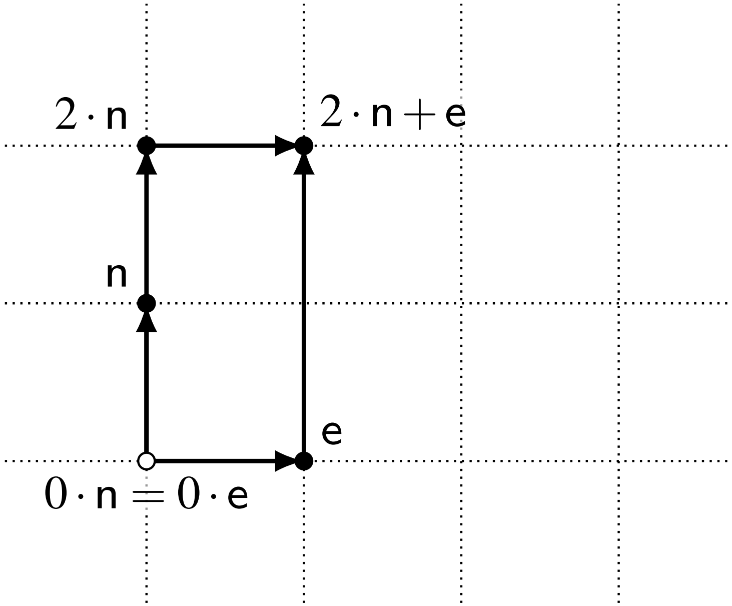
GBF

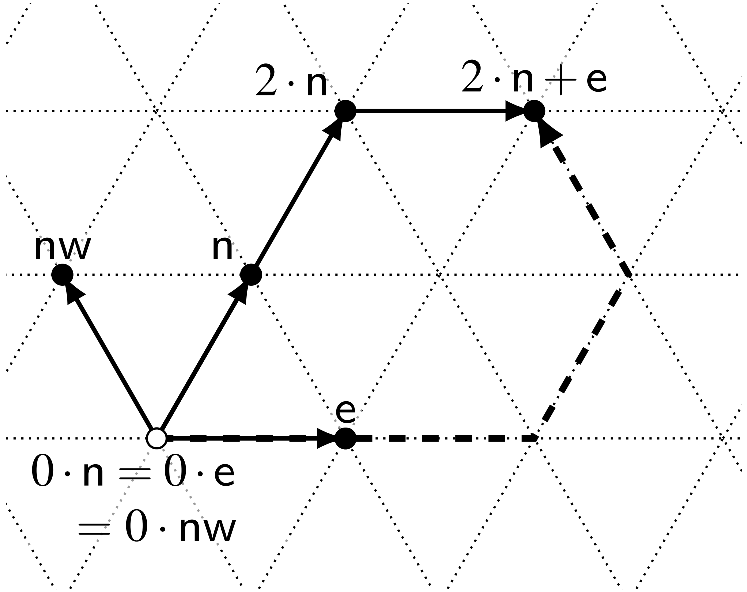


Equations are closed paths (loops):

- backtracking path are closed in any Cayley graph : e + e + n - n - e - e
- group equation are specific of the graph topology

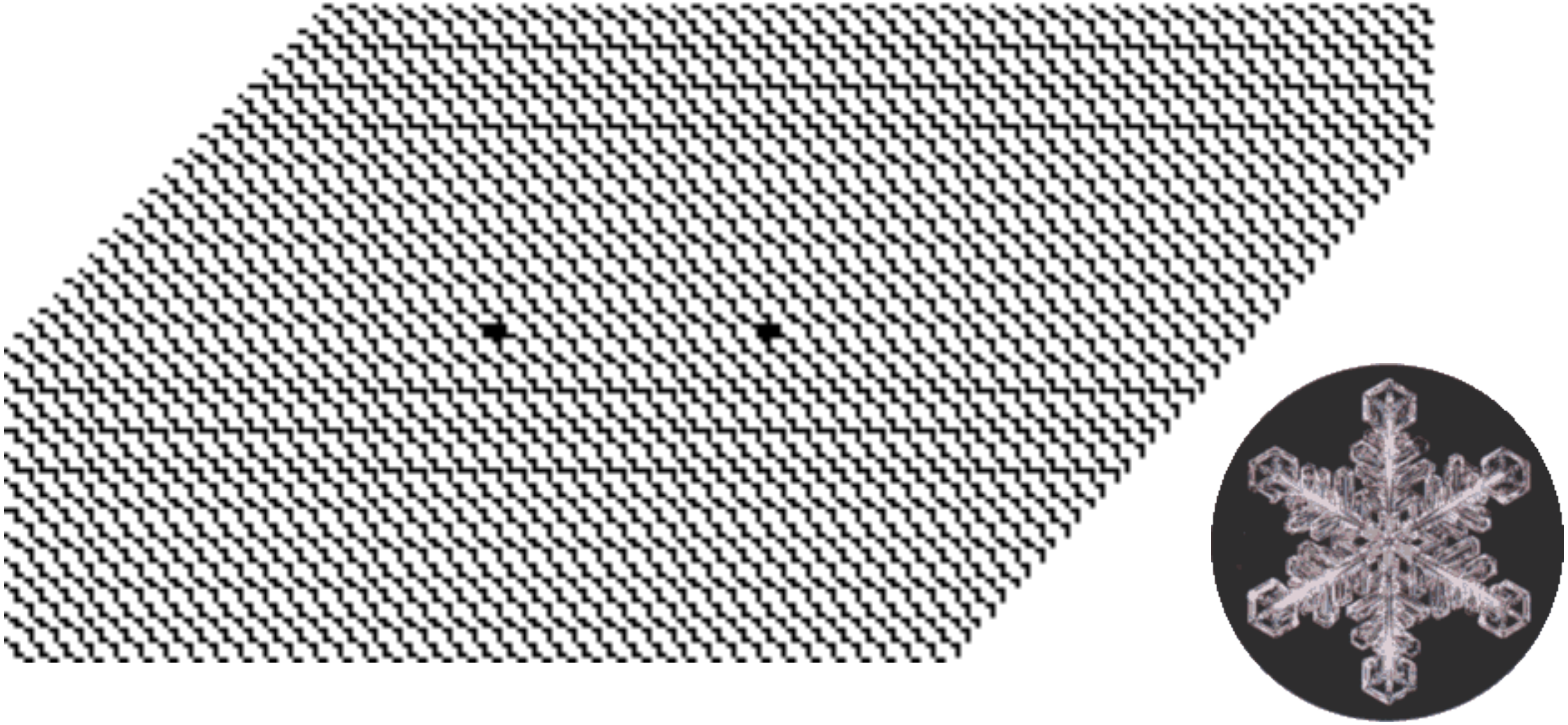
GBF





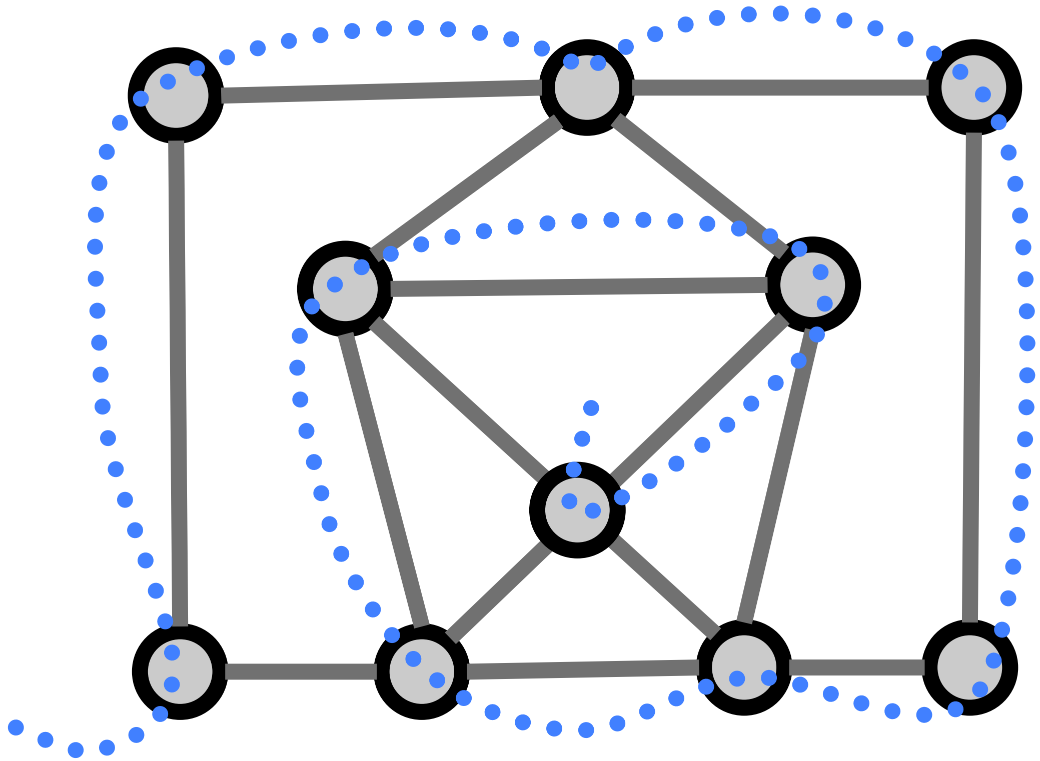
< n, e, nw; n = e+nw >

Snowflake formation (CA-based)



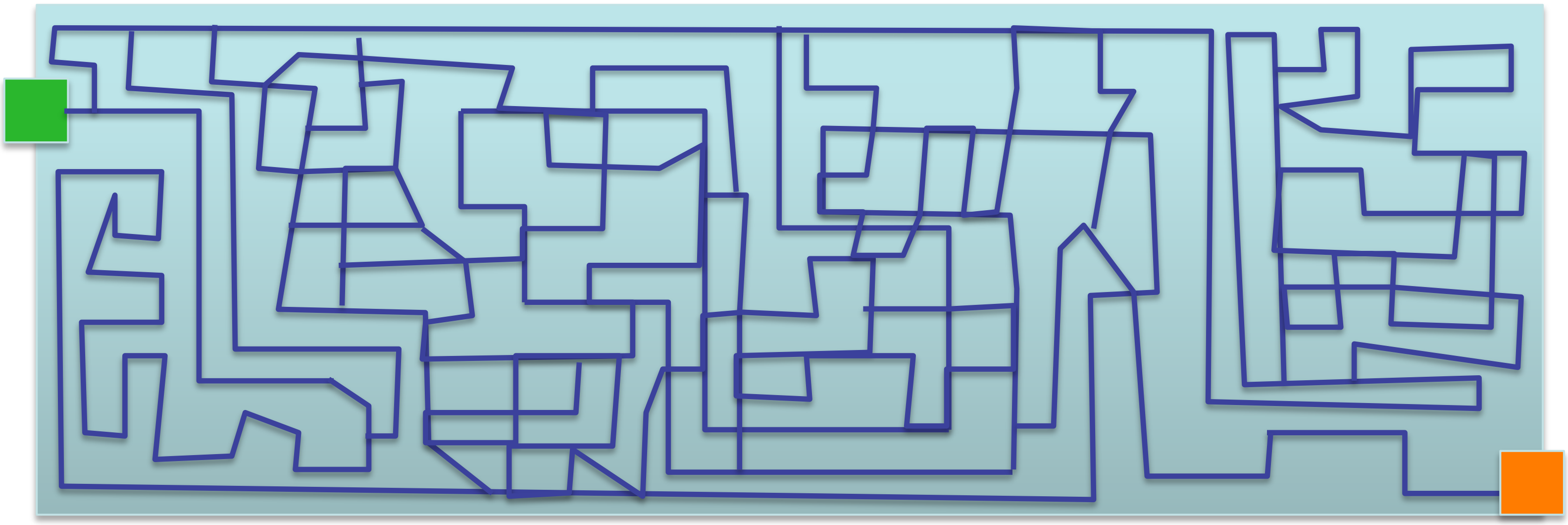
```
gbf hexa = <a, b, c;  a+b=c>

trans T = {
   (0 as x / (neighborsfold(+, 0, x)==1)
    => 1)
}
```

Hamiltonian path



```
trans hamiltonian_path = {
        x* as p / size(p) = N => return(p)
 };;
```

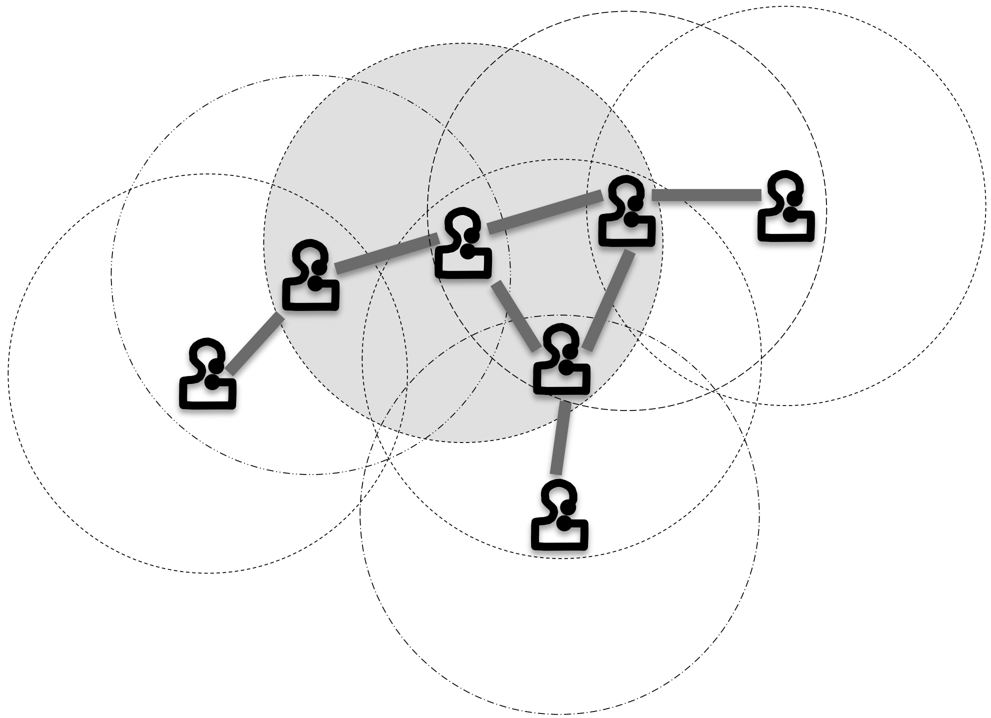
Path in a Maze



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

Graphs defined by a metric

Proximal

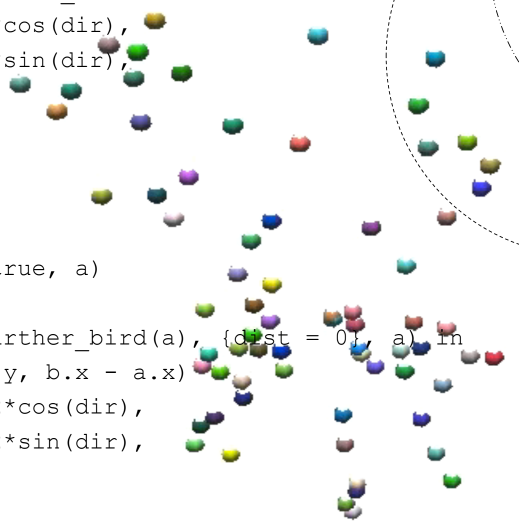


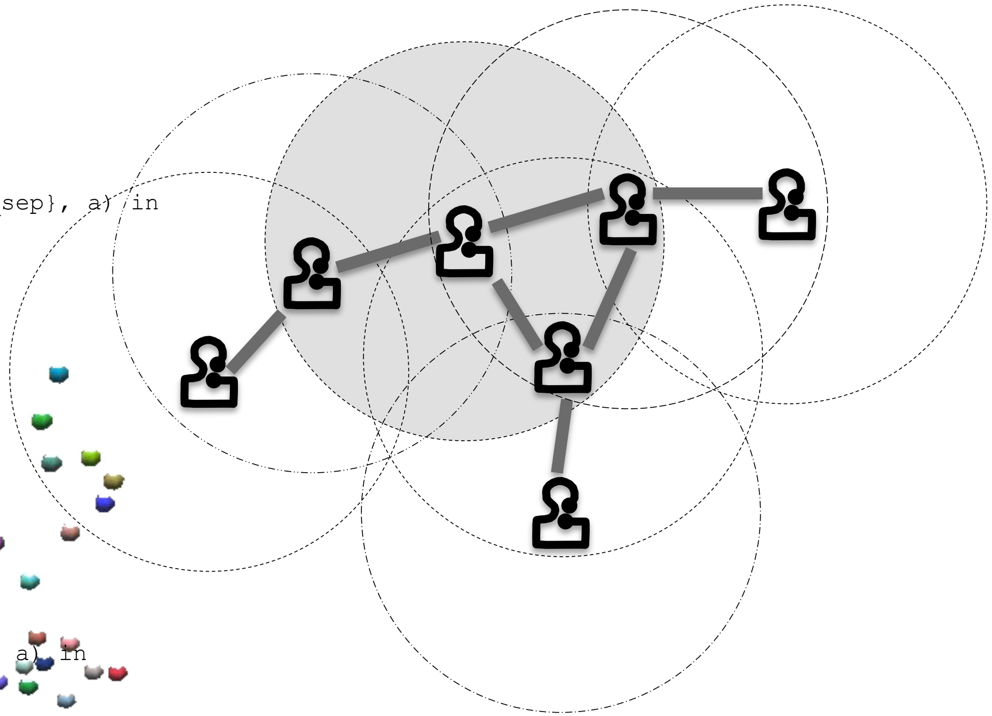
record agent = { x : float, y: float }

proximal P[agent] = fun a b -> (a.x - b.x)^2 + (a.y – b.y)^2

Proximal and Flocking Birds

```
trans Flocking = {

  separation =
    a / neighborsfold(to_close(a), false, a)
    => begin
          let b = neighborsfold(closer_bird(a), {dist = 2*d_sep}, a) in
          let dir = if (random(2) == 0)
                    then b.theta + 'PI_2
                    else b.theta - 'PI_2 fi
          in a + {x = a.x + speed*cos(dir),
                  y = a.y + speed*sin(dir),
                  theta = dir}
       end;


  cohesion =
    a / neighborsfold(to_far(a), true, a)
    => begin
          let b = neighborsfold(farther_bird(a), {dist = 0}, a) in
          let dir = atan2(b.y - a.y, b.x - a.x)
          in a + {x = a.x + speed2*cos(dir),
                  y = a.y + speed2*sin(dir),
                  theta = dir}
           end;

  alignment =
    a => begin
            let phi = neighborsfold(add_theta, 0, a)
            and nb = neighborsfold(nb_neighbors, 0, a) in
            let dir = phi / nb
            in a + {x = a.x + speed*cos(dir) + random(bruit),
                    y = a.y + speed*sin(dir) + random(bruit),
                    theta = dir}
         end;
};;
```





```
separation =
  a / neighborsfold(to_close(a), false, a)
  => begin
        let b = neighborsfold(    closer_bird(a),
                                  {dist = 2*d_sep},
                                  a ) in
        let dir = if (random(2) == 0)
                  then b.theta + 'PI_2
                  else b.theta - 'PI_2 fi
        in a + {x = a.x + speed*cos(dir),
                y = a.y + speed*sin(dir),
                theta = dir}
     end;
```

```
cohesion =
  a / neighborsfold(to_far(a), true, a)
  => begin
        let b = neighborsfold(closer_bird(a),{dist = 0}, a) in
        let dir = atan2(b.y - a.y, b.x - a.x)
        in a + {x = a.x + speed2*cos(dir),
                y = a.y + speed2*sin(dir),
                theta = dir}
     end;
```

```
alignment =
  a => begin
          let phi = neighborsfold(add_theta, 0, a)
          and nb = neighborsfold(nb_neighbors, 0, a) in
          let dir = phi / nb
          in a + {x = a.x + speed*cos(dir) + random(noise),
                  y = a.y + speed*sin(dir) + random(noise),
                  theta = dir}
       end;
```

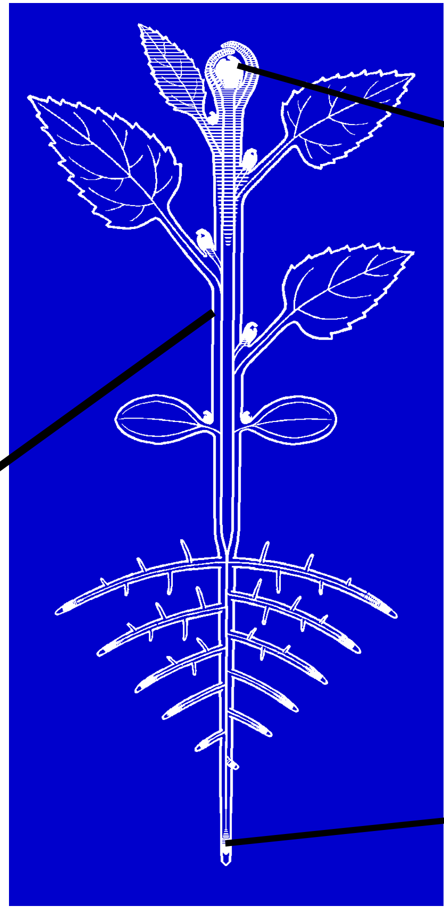
The Growth of a Meristem

[PNAS 103(5), 1627-1632, 2006]

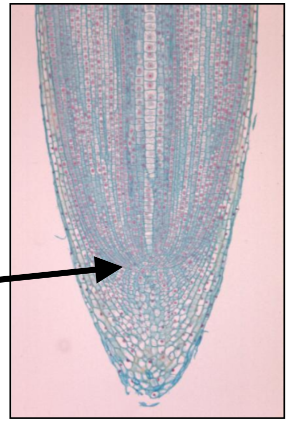
Pierre Barbier de Reuille
Mikaël Lucas
Jan Traas
Christophe Godin
CIRAD/INRA/INRIA

Organs positionning at the shoot apex

Cambium

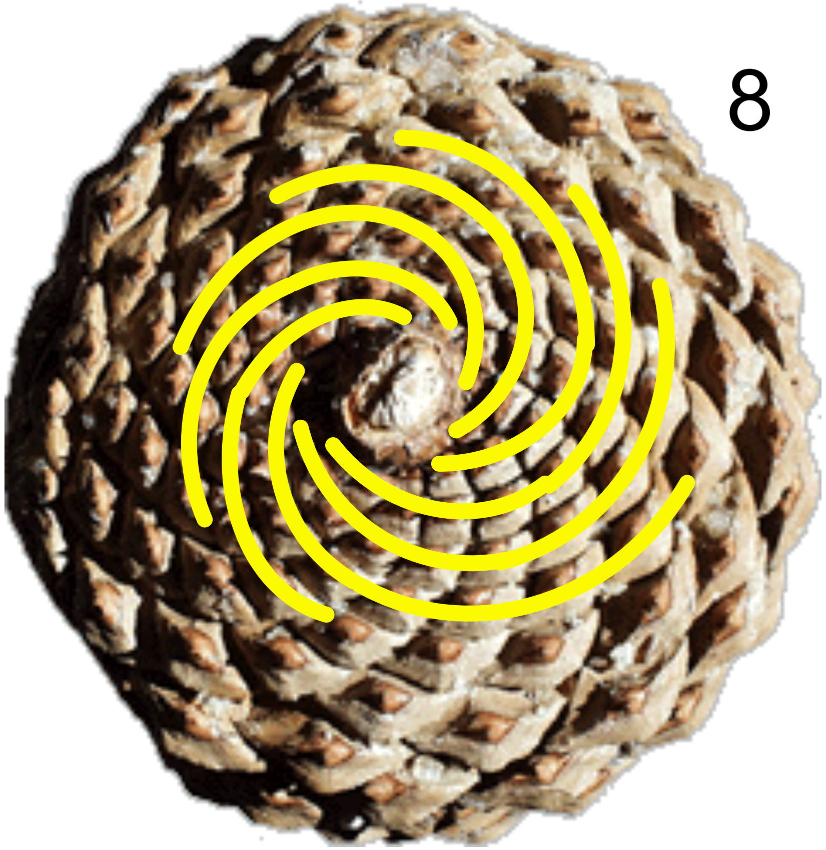


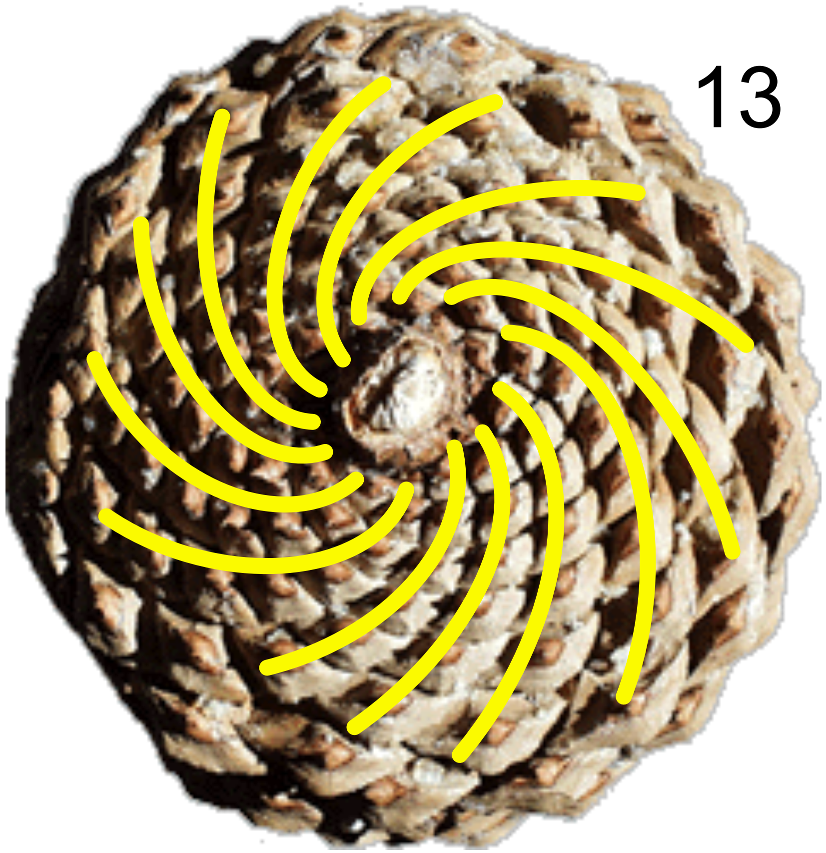
Shoot apical meristem



Root apical meristem

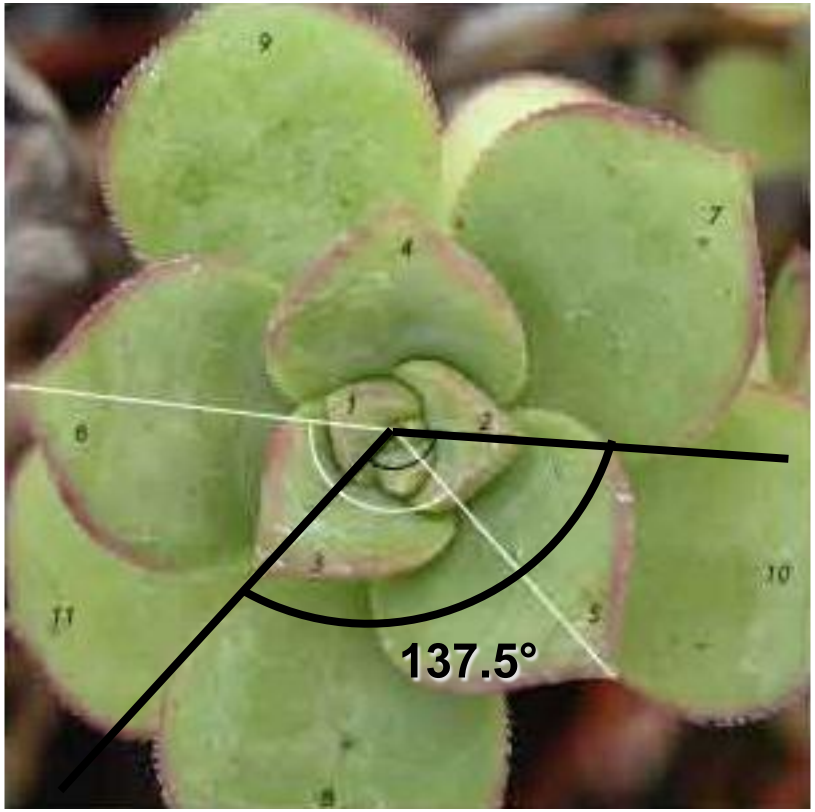
Fibonacci and phyllotaxis





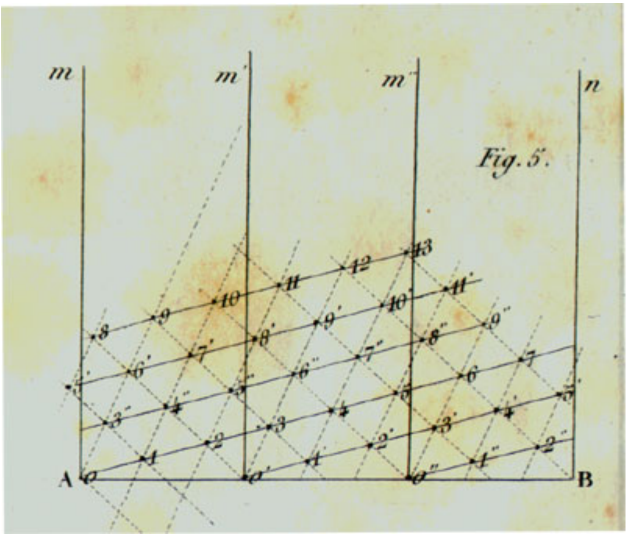
8,13 → Two successive numbers of the Fibonacci series

Phyllotaxis : divergence angle



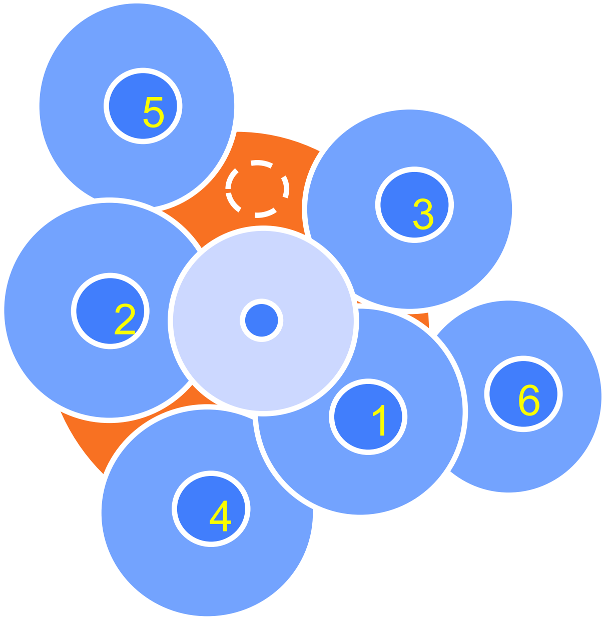
Phyllotaxis models: three kinds of approaches

Geometrical



(Bravais & Bravais,1837)

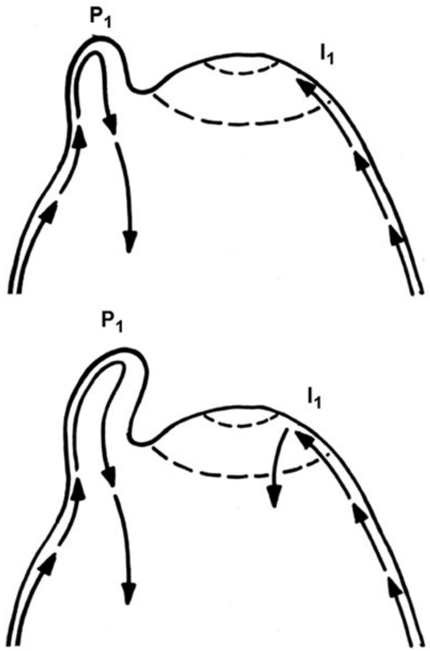
Dynamical



(Hofmeister, 1868)

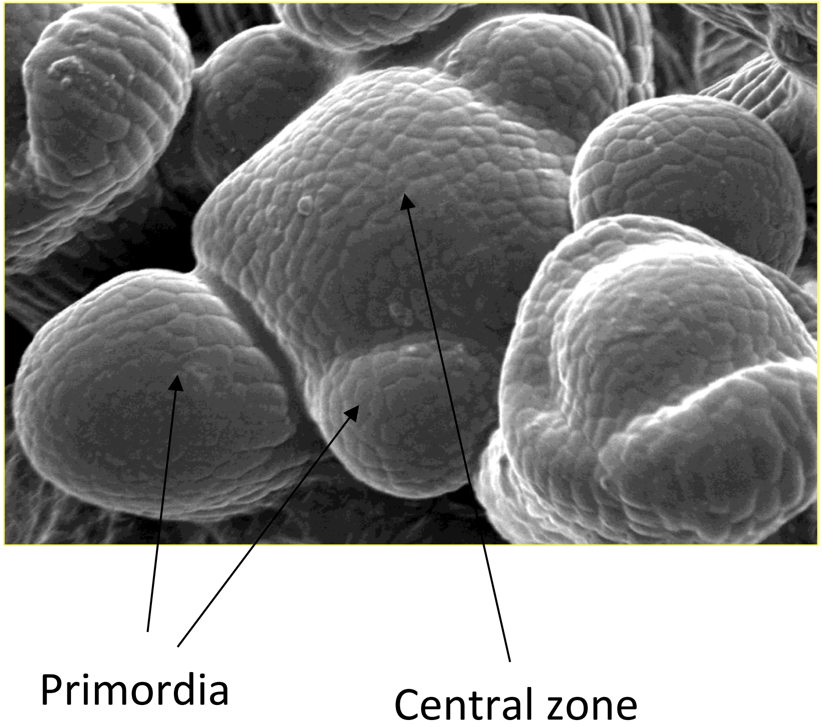
(Snow and Snow, 1962)

Physiological



(Reinhardt et al., 2000)

A shoot apical meristem



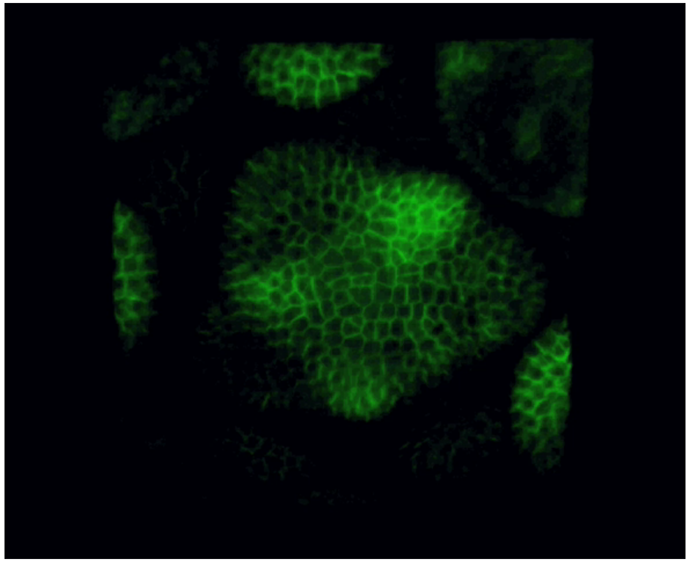
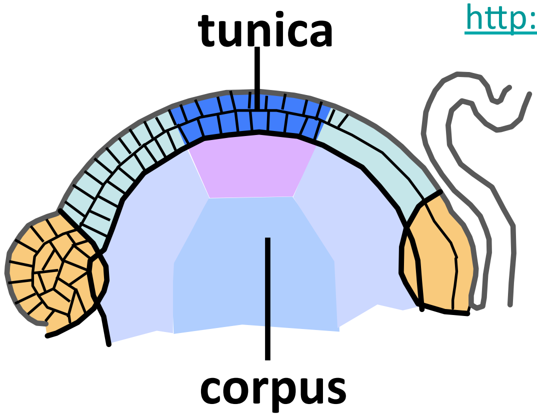


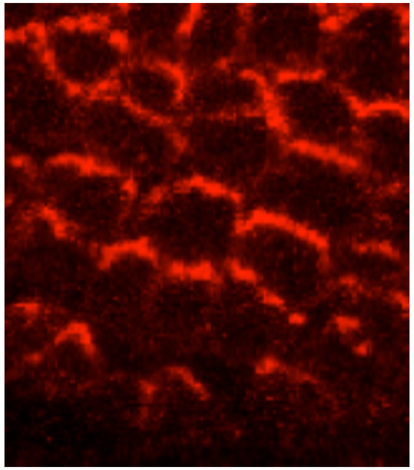
Image sequence showing cell division patterns via membrane-bound PIN1, in Shoot Apical Meristem (SAM), nearby floral meristems, and the boundaries between them (M. Heisler).
http://computableplant.ics.uci.edu/ (E. Mjolness)



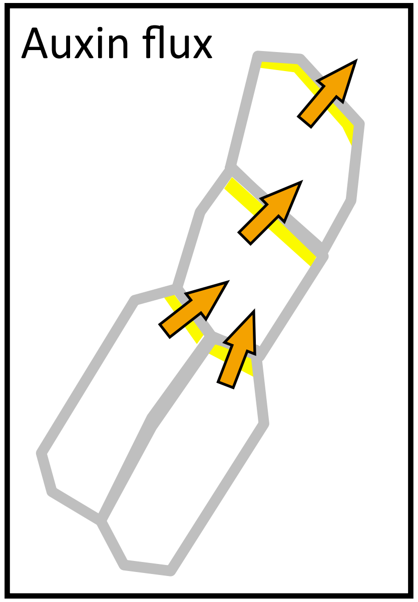
Active transport of auxine

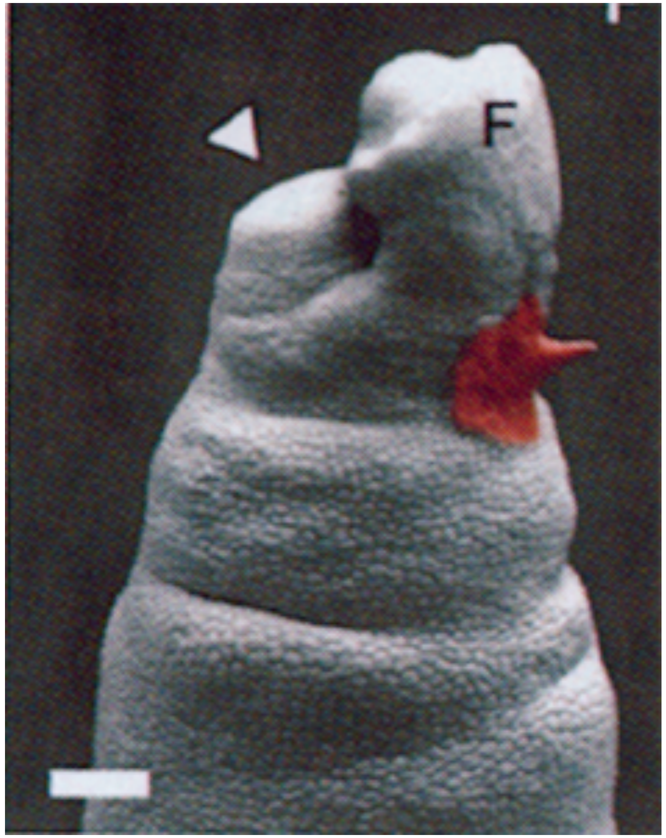
wild type

pin-1 mutant



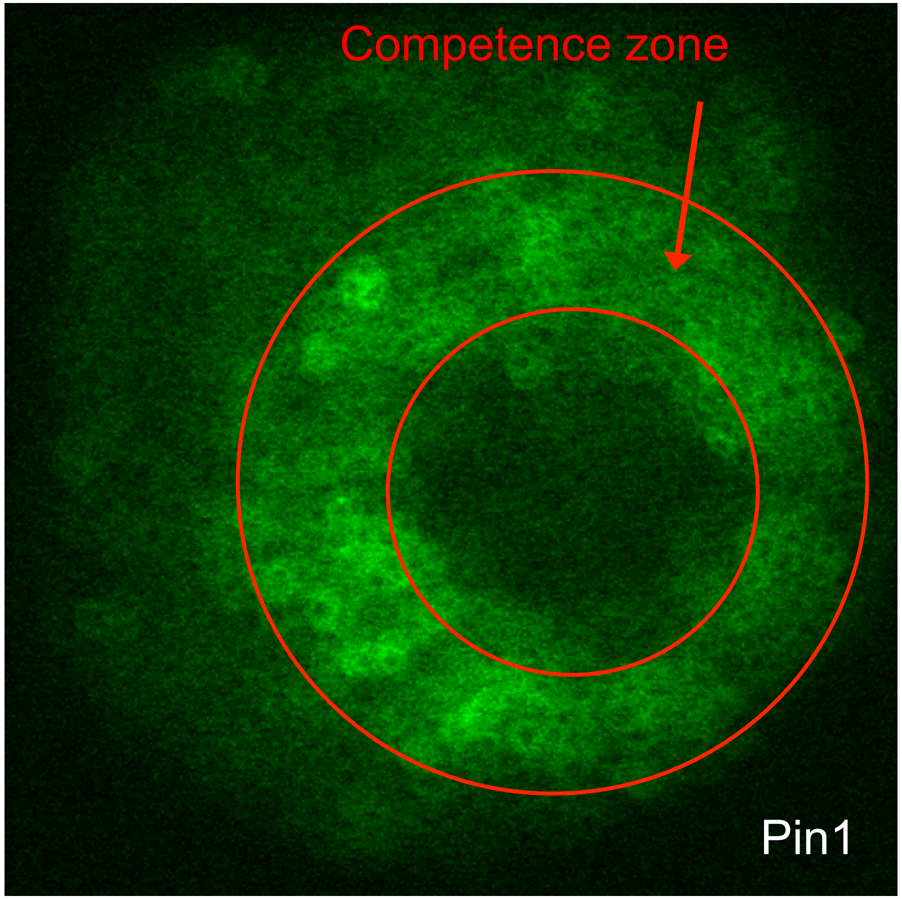
Immunolabelling of
PIN-FORMED1 protein





high concentration of
auxine induces organ initiation

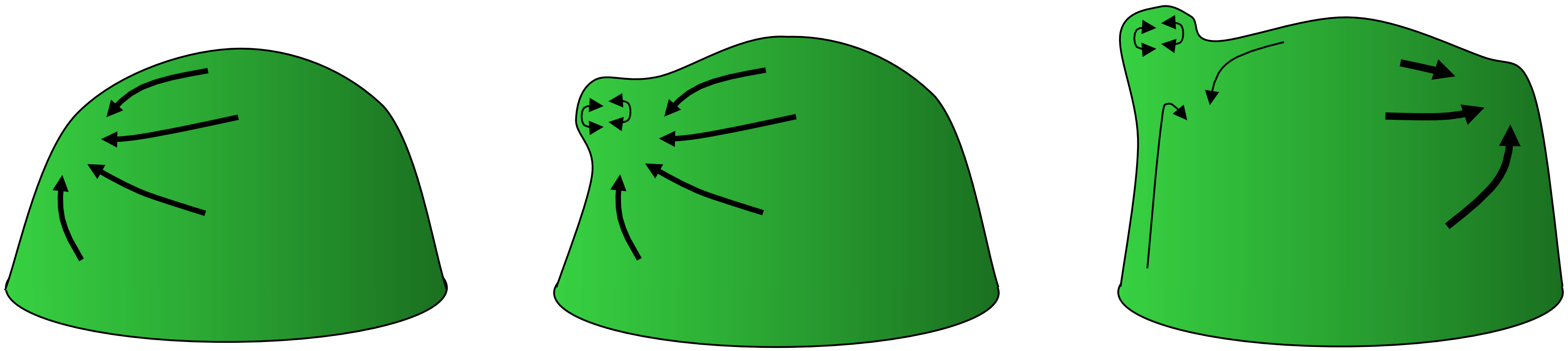
Genetic labelling



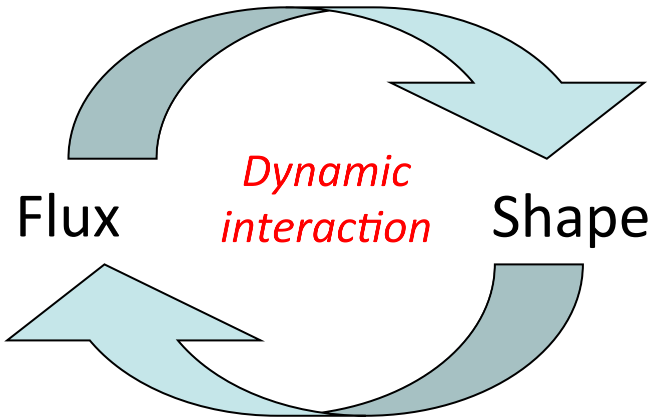
Images : Vernoux & Traas

ANT::GFP

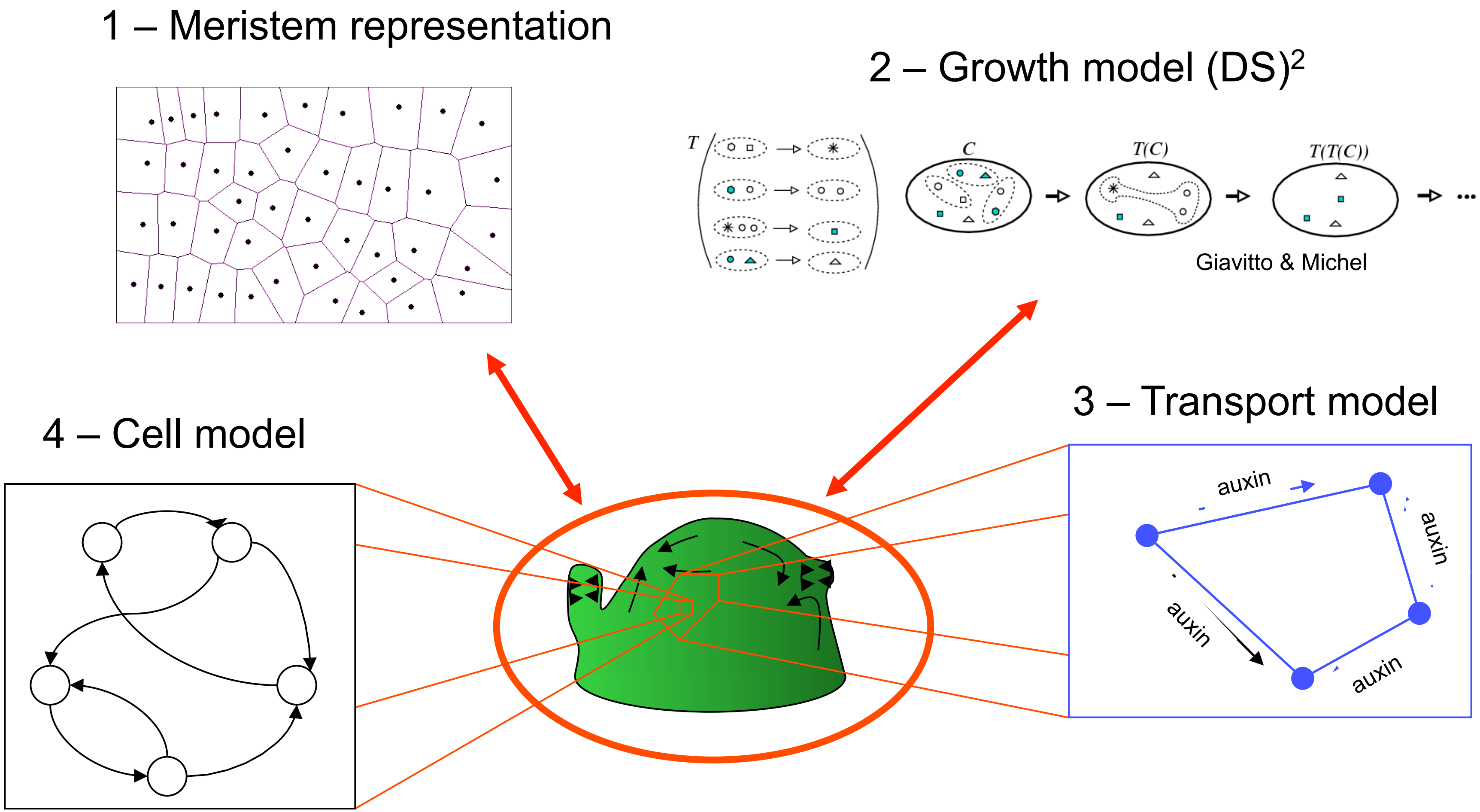
(DS)²



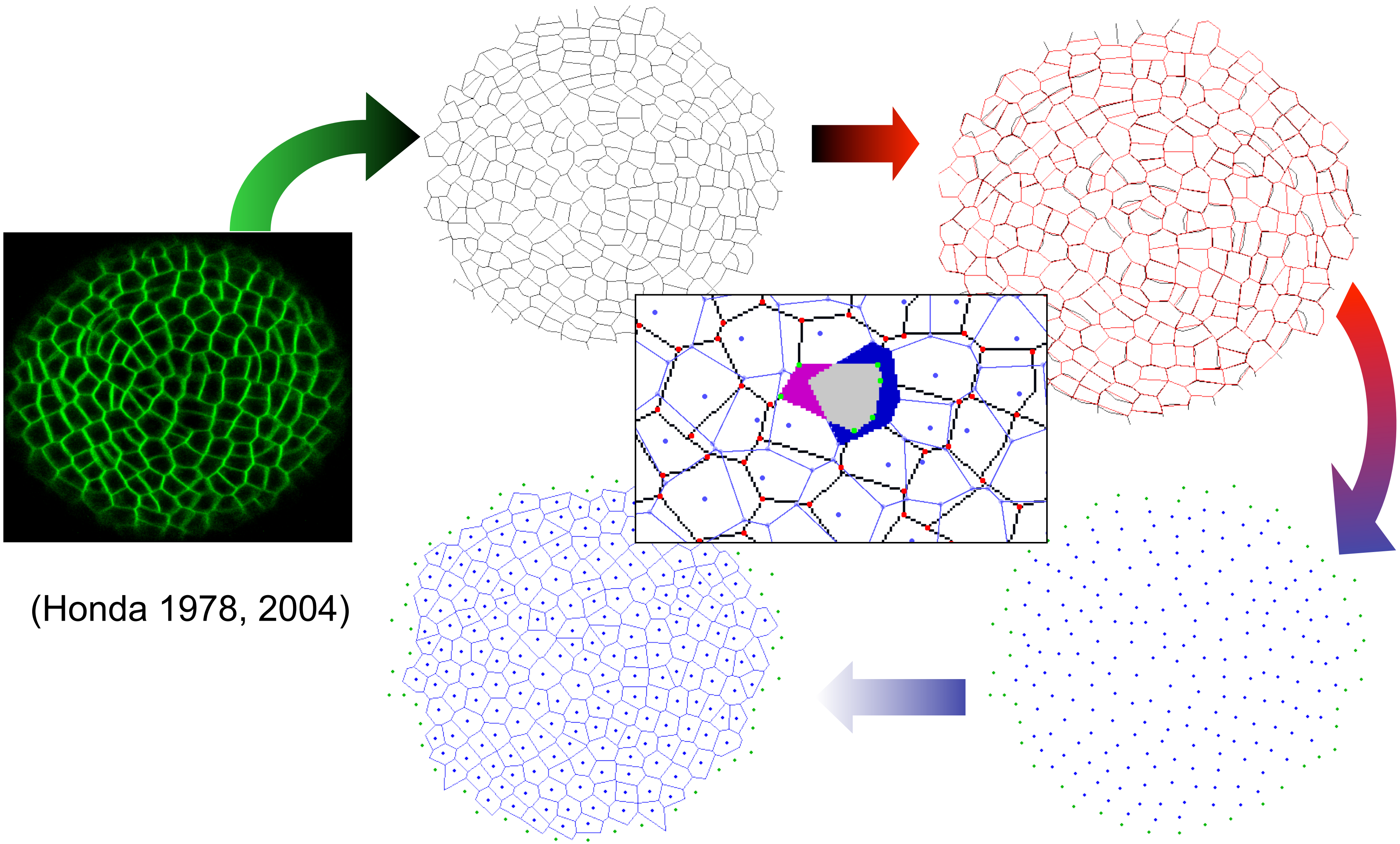
flux…   changes form…   which changes flux…



Virtual meristem



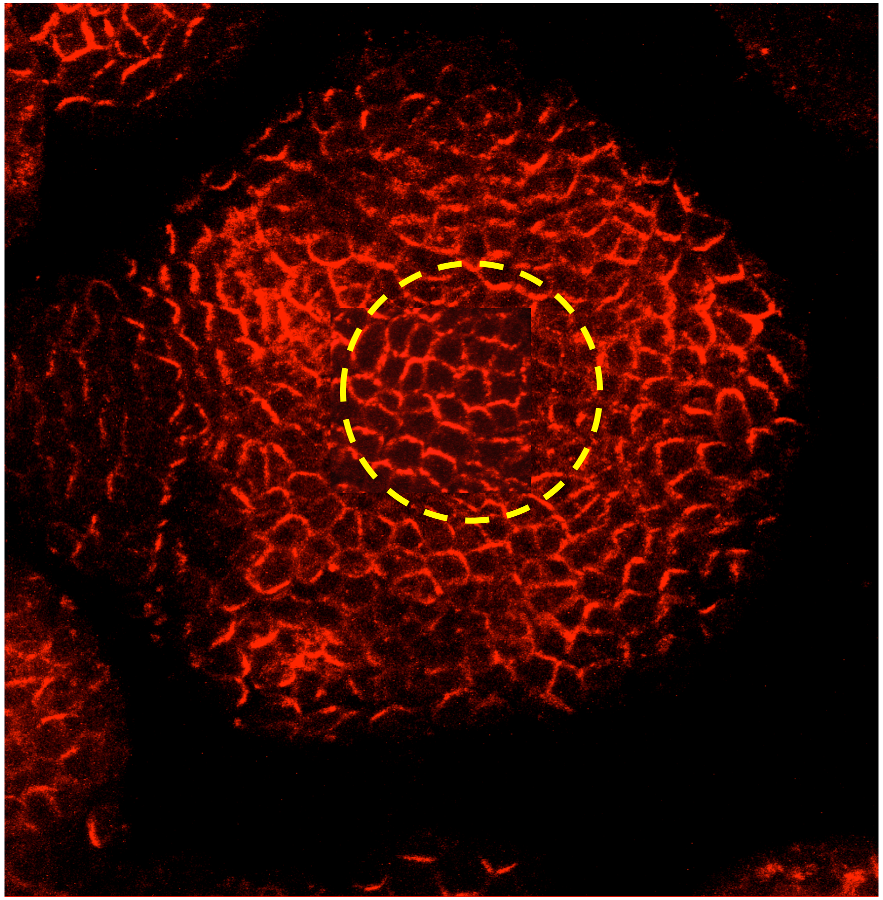
Meristem representation



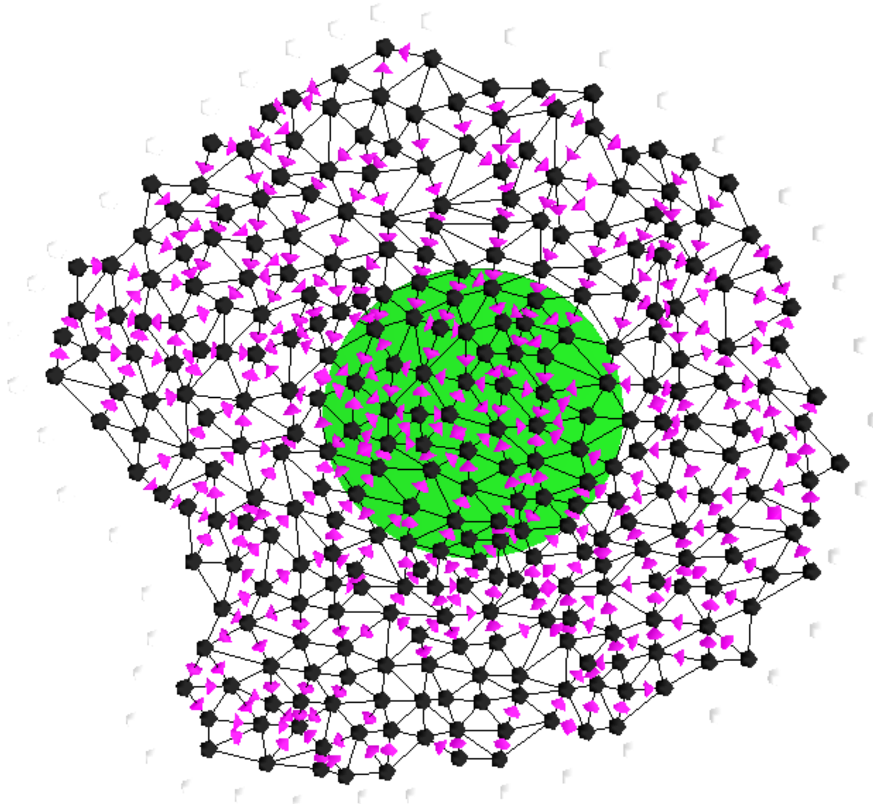
(Honda 1978, 2004)

(Barbier de Reuille et al. 2003)

Meristem representation



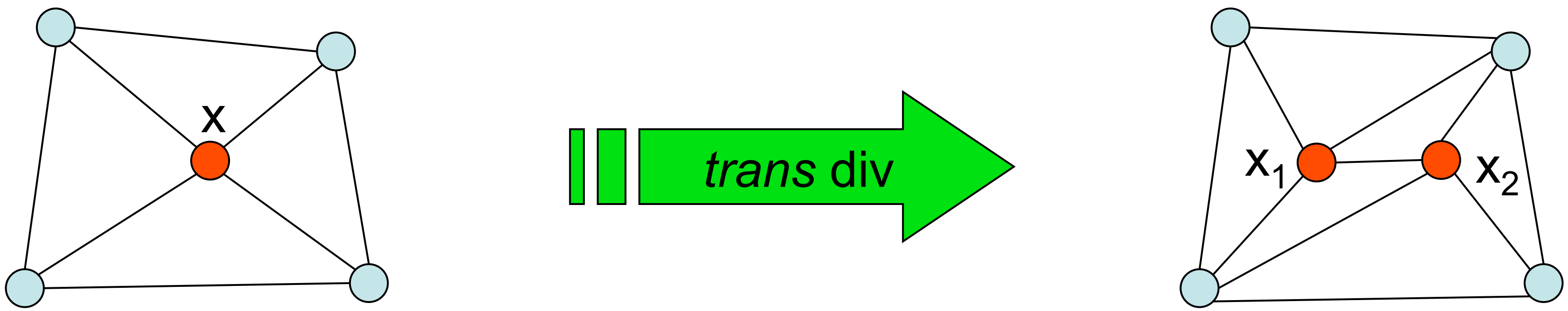
Original



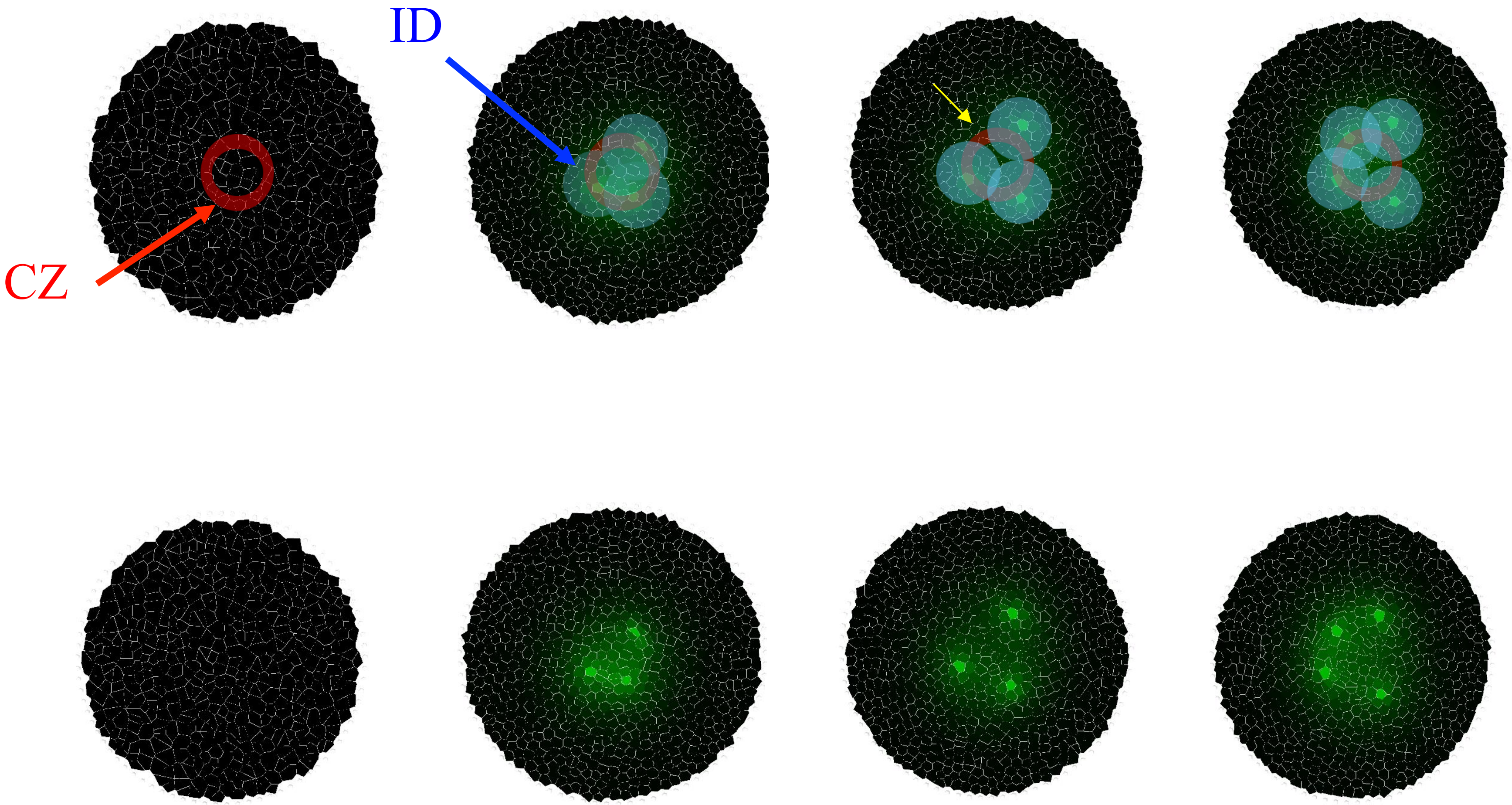
Digitalized

MGS – dynamic structure rules

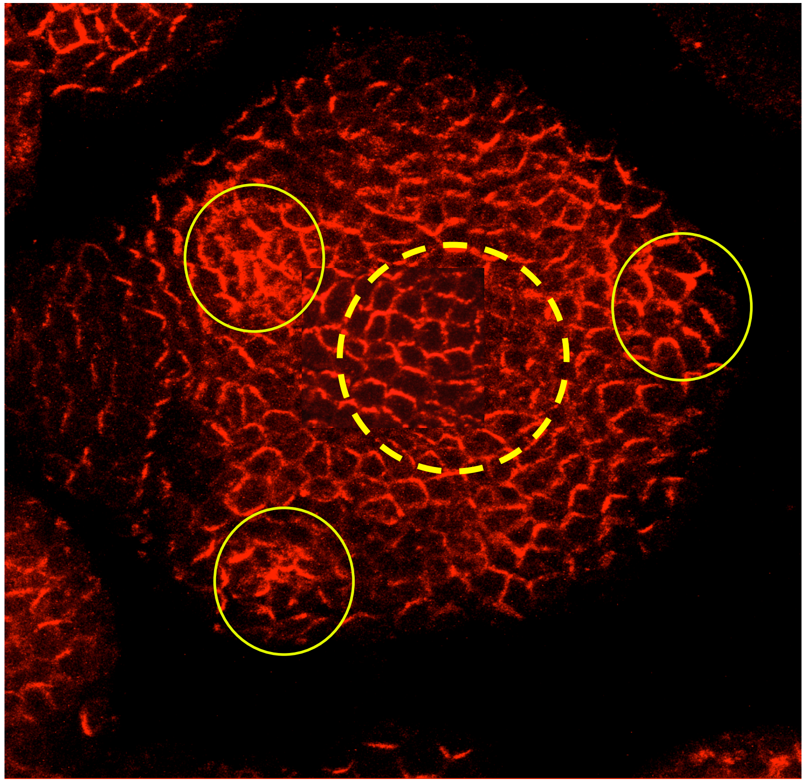
$$\textit{trans}\ \mathrm{div} = \left\{ x / \textit{dividing}(x) \Rightarrow \textit{child}(x,1), \textit{child}(x,2) \right\}$$

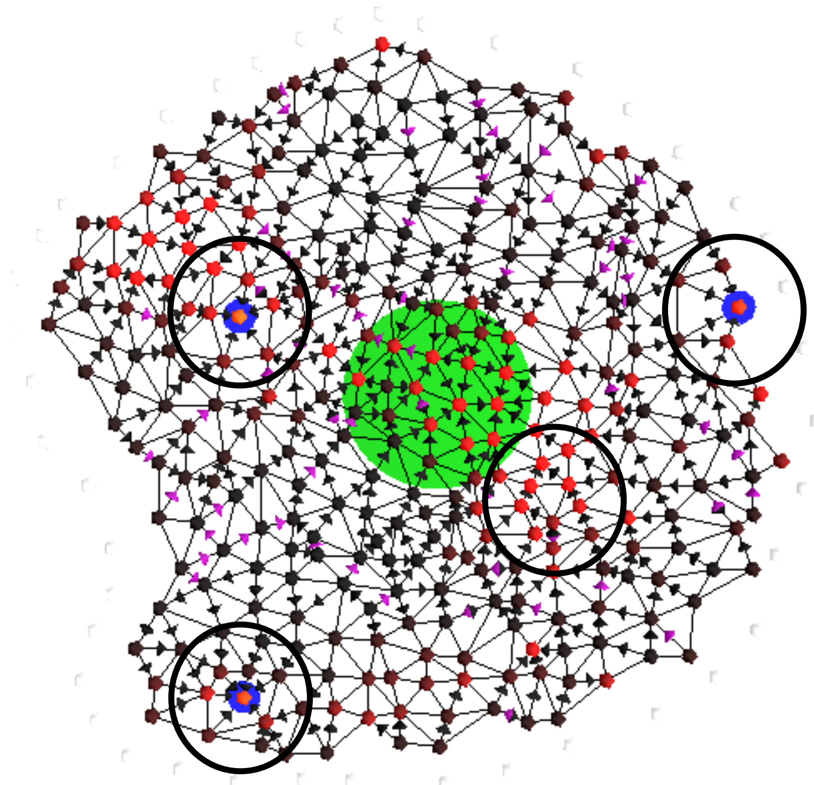


Model 3 - “Inhibitor fields” and diffusion



Simulation results





Model 4 : Active pumping of auxin

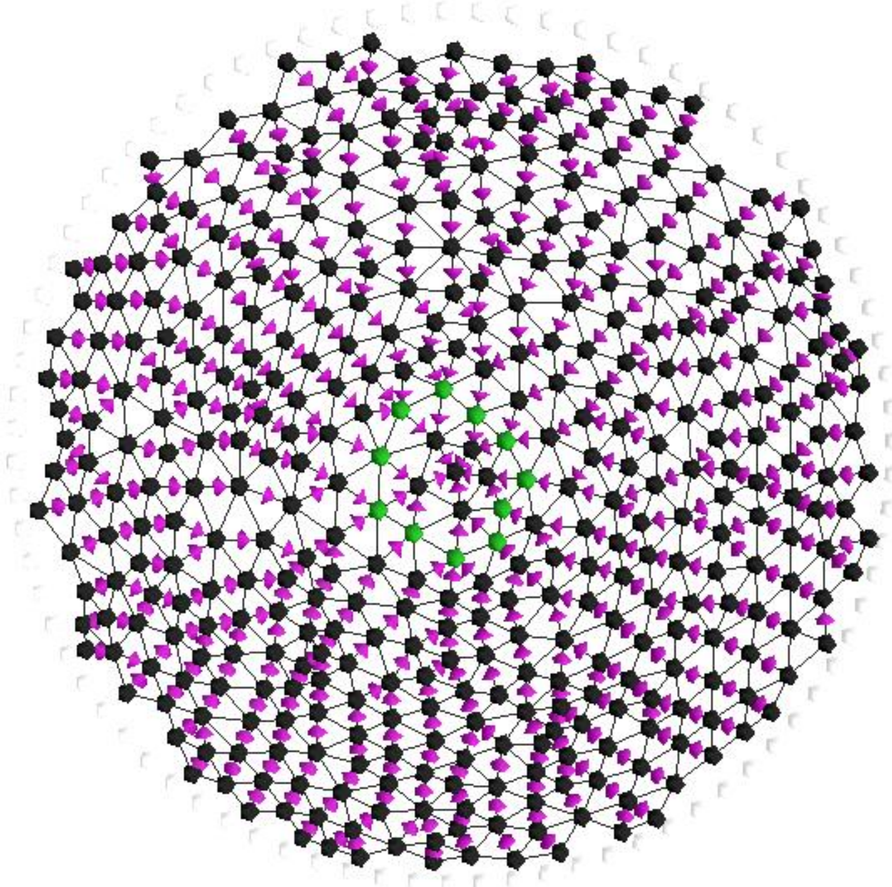
- *Cell internal state and processes =>*
  - capacity of division,
  - spring relaxed length,
  - primordium/center,
  - **concentration of auxin (inhibitor),**
  - **saturation,**
  - **auxin degradation / evacuation**
  - **promotion to primordium**
  - **“pump magnetism”**

- *Movement =>*
  - due to cell growth

- *Growth =>*
  - increase of spring relaxed length

- *Division =>*
  - when size > threshold

- *Cell interaction =>*
  - **Passive diffusion of auxin,**
  - **active pumping of auxin**



Model

- ***Cell internal state and processes***
  capacity of division, springs relaxed length, primordium/center,
  concentration of auxin, auxin degradation / evacuation, inhibitor
  promotion to primordium, “pump magnetism”

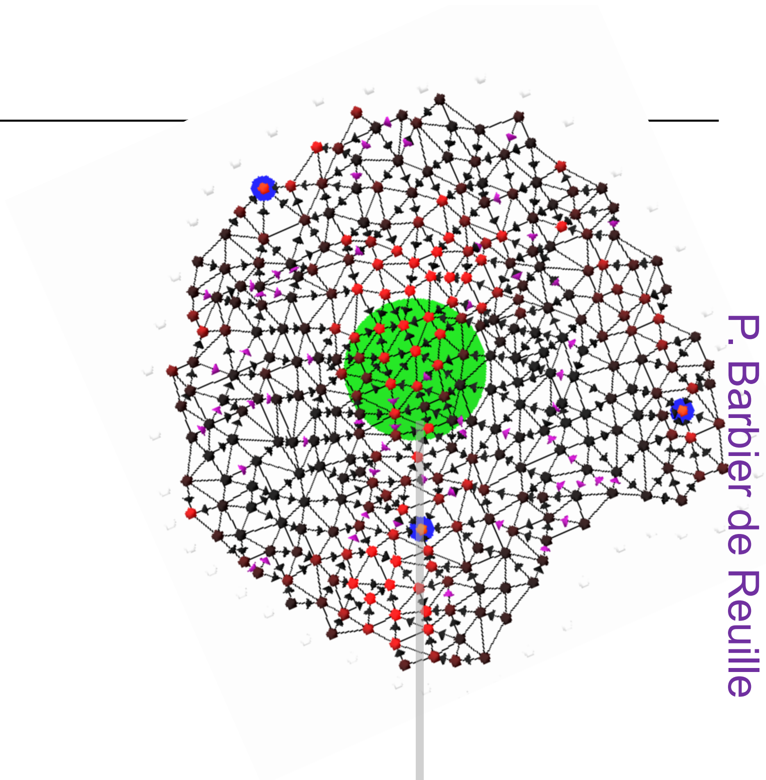
- ***Movement*** (due to cell growth)

- ***Growth***: increase of spring relaxed length

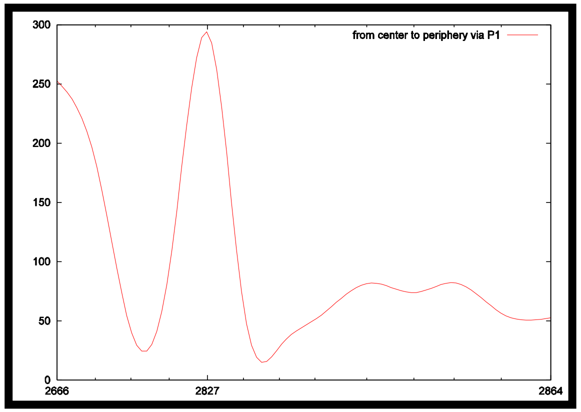
- ***Division***: when size > threshold

- ***Cell interaction***
  Passive diffusion of auxin, active pumping of auxin

```
trans Auxin = {
        x, y / pump(x,y)
        → x+{x.auxin -= δ}, y+{y.auxin += δ}
}
```

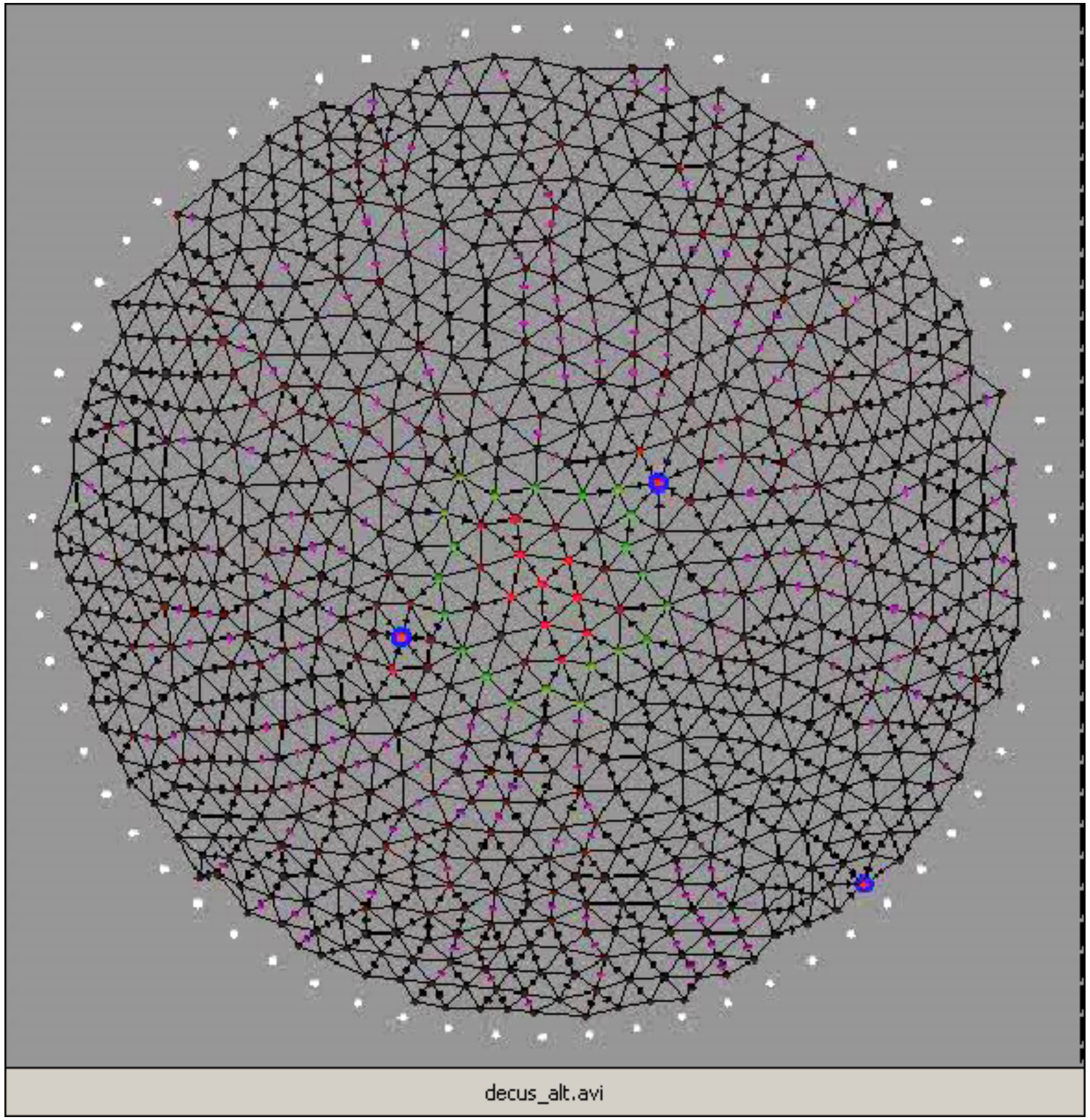


P. Barbier de Reuille

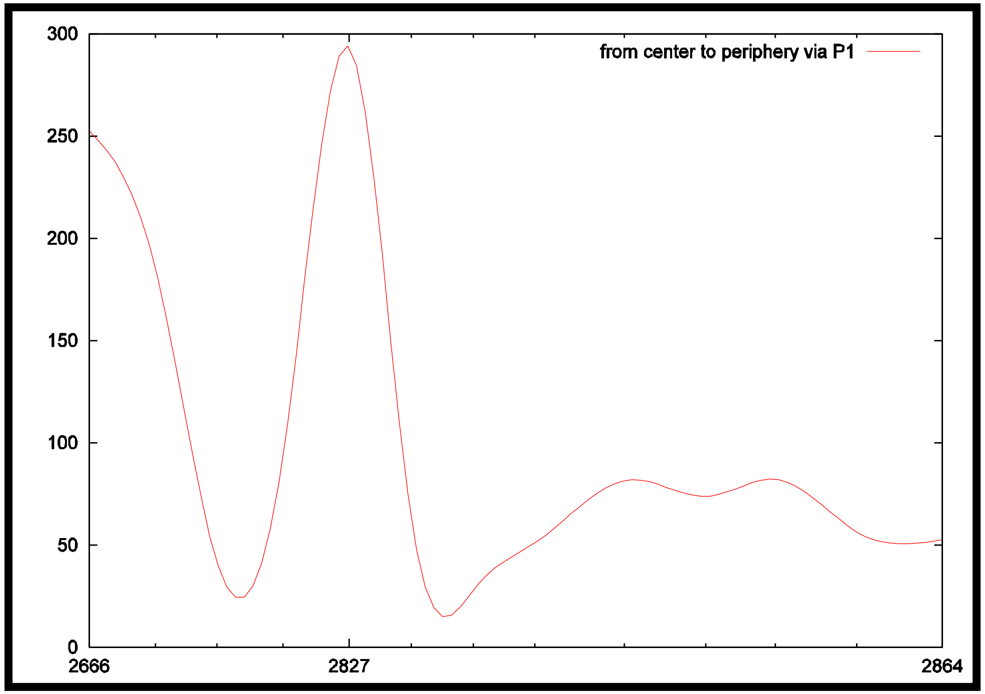


Auxin level

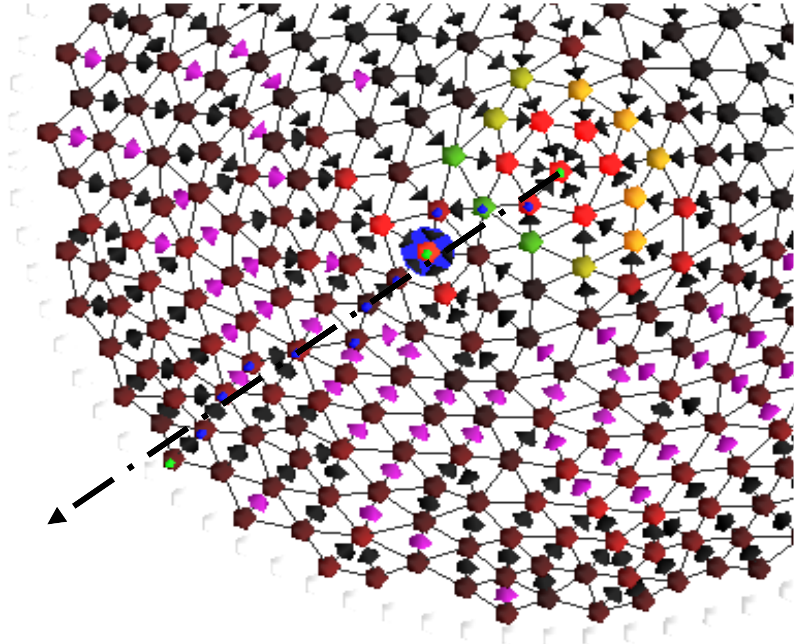
Simulation



Primordium local inhibition



Auxin level



Acknowledgements

MGS: Antoine Spicher, Olivier Michel, Julien Cohen

S&S Bio : Hanna Klaudel, Franck Delaplace, Hugues Berry, Przemek Prusinkiewicz, Annick Lesne…

Spatial Computing: Jacob Beal, Fréderic Gruau, René Doursat…

Examples: Pierre Barbier de Reuille, Christophe Godin, Samuel Bottani, the Paris iGEM'07 team…

Some figures are borowed from Olivier Michel, Antoine Spicher, Pierre Barbier de Reuille, Franck Delaplace, Hugues Berry (INRIA), the iGEM Paris 2007 and many others.