A Computational Framework for Modelling (and Simulating) Multicellular Biochemistry

Sara Montagna Mirko Viroli

{sara.montagna, mirko.viroli}@unibo.it

Alma Mater Studiorum—Università di Bologna a Cesena

IEEE Congress on Evolutionary Computation (CEC 2009)
Trondheim, Norwey, 18th–21st May 2009

Outline

1. On modelling and simulating the morphogenesis of living systems
   - open problems, existing tools and their inadequacy
2. A new simulation framework
   - computational model
   - specification language
   - simulation engine
3. A case study of *Drosophila Melanogaster*

Outline

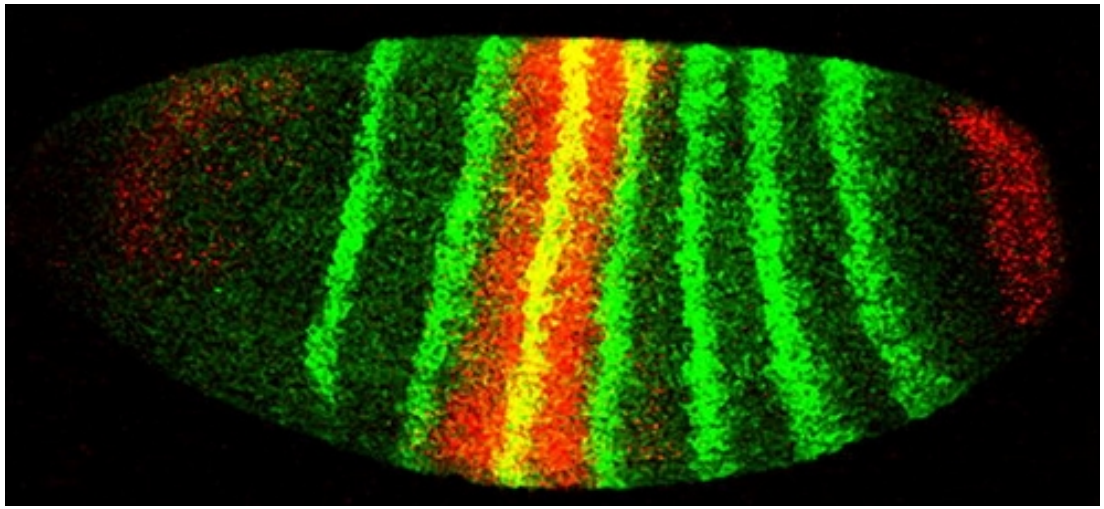
Biological Background

Developmental Biology researches the mechanisms of development, differentiation, and growth in animals and plants at the molecular, cellular, and genetic levels.

Animal developmental steps

1. Fertilisation of one egg
2. Mitotic division
3. Cellular differentiation
   - diverse gene expression
4. Spatial organisation of the cell diversity
   - each region of the developing organism expresses a given set of genes

Facing a very complex problem



The emergence of embryonic patterning – self-organized structures

- Role of transcriptional control mechanisms
- Role of signalling pathways
- Role of cell-to-cell direct interaction
- Role of short and long range signals (*morphogenes*)

$\longrightarrow$ interplay between cells internal activity and cell-to-cell interactions

On the need of proper modelling and simulation tools

Are simulation tools ready for this scenario?

- We need them even before mechanisms are understood
- This is key to properly evolve nurturing ideas and check hypotheses

Requirements

1. Multi-compartment model
   - for modelling interactions between the parts of the cell..
   - ..but also of the many cells in the systesm
2. Exact chemical dynamcis
   - for properly capturing chemical evolution and its self-organization
3. Chemical transfer
   - for studying the effects of short and long range signals/interactions

Brief Survey on Existing Frameworks

Systems Biology Tools: see http://systems-biology.org/software/

Based on classical mathematical models (ODE or PDE)

- CellDesigner, CellWare, COPASI, Dizzy, JDesigner, Virtual Cell, ...
- Mainly thought for intracellular aspects (biochemical pathways)

Computational Systems Biology Tools

Based on computational models (stochastic process-algebras, Petri-Nets)

- ground on Gillespie's characterisation of chemistry as CTMC
- SPIM (stochastic π-calculus), BlenX, Bio-PEPA
- apply a "concurrent programming language approach" to modelling
- promote a view of "molecules as processes"

They do not scale well with the system complexity (e.g. number of cells)

Outline

A new modelling and simulation framework

Motivations

Tackle scenarios of dev. biology..

- Naturally supporting scenarios with many compartments
- Use state-of-the-art implem. techniques for the simulation engine

..using pros of the two scenarios

- Focussing on chemical reactions rather than molecules-as-protocols
- Using a "programming language" to configure complex simulations

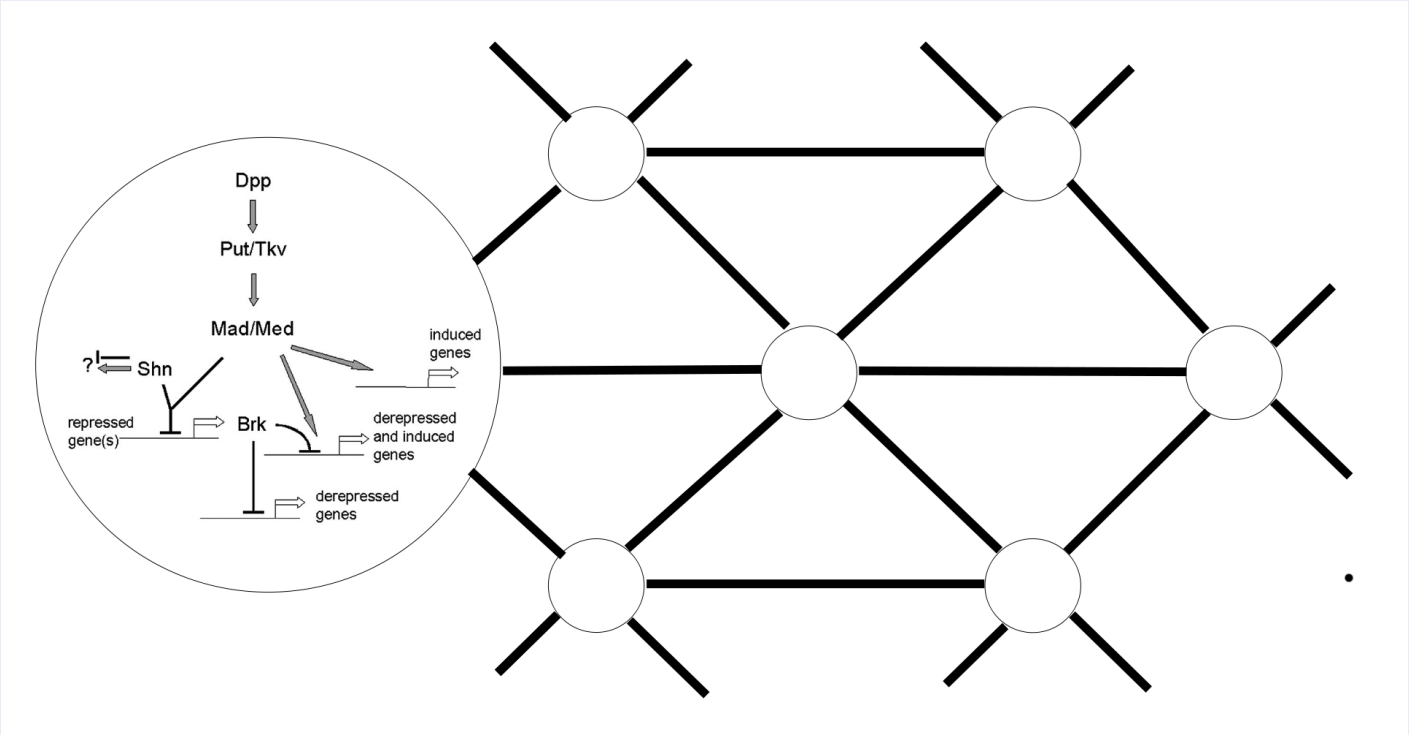
State of development

- A functionally-complete prototype is internally available
- A public version for the community planned in few months
- Structure is analogous to existing tools like SPIM
  - front-end compiler, simulation engine, graphical tools are not included

Level 1: the Computational Model – Structure

A multi-compartment version of standard CMSB view

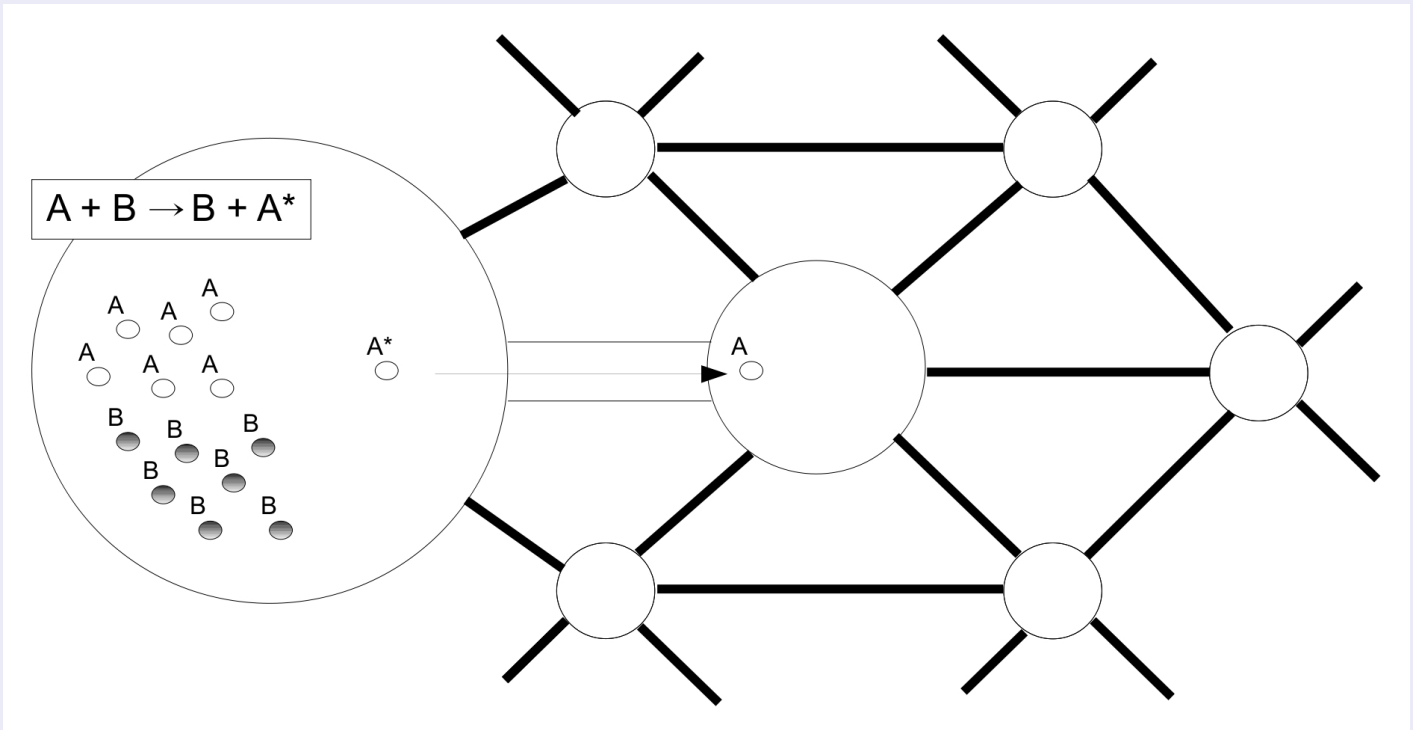
- A system is a graph-like network of compartments
- Each compartment hosts a chemical solution (a simple pathway)
- Each chemical reaction enables a Markovian transition [Priami, 1995]
- Simulation computes the concentration of reactants over time



Level 1: the Computational Model – Chemical Transfer

Transfer Model

- Some chemical reactions can produce so-called firing molecules
- They are sent to a neighbouring compartment picked probabilistically
- (mobility and mitotic division will be supported in future versions)



Level 2: the Surface Specification Language

Structure

A program as a set of logic declarations

- Declaring molecules, reactions, compartments, links
- Configuring the simulation (initial state and parameters)
- Stating output commands

Declarations can have variables and be equipped with fully expressive preconditions, acting as constraints on declarations

- Supporting flexibility

Surface Language Constructs

Declaration constructs provided by the language

```
const K.                                % K is a fact
molecule M.                             % molecule with id M
reaction T : Li --> Lo rate R.          % chemical reaction with id T
compartment C.                          % compartment with id C
link Cs >>> Cd rate R molecule M.       % link for M, from compartment Cs to Cd
concentration N of M in C.              % N molecules of M in compartment C
place T into C.                         % places reaction T in compartment C
final_time R.                           % simulation time
final_steps N.                          % simulation steps
sample_time R.                          % time between two observations
sample_steps R.                         % steps between two observations
out molecule(C,M).                      % print a molecule concentration
out time.                               % print elapsed time
out step.                               % print number of steps so far
out string(T).                          % print a string
..
```

Example specification

Diffusion of a substance into a grid-like tissue

```
const size(20).

molecule M where (M in [pump,field]).
reaction r(pump)  : [pump] --> [pump,field] rate 10.0.
reaction r(diff)  : [field] --> [field,firing(field)] rate 0.2.
reaction r(decay) : [field] --> [] rate 0.1.

compartment c(X,Y) where (const size(N), X in {1..N}, Y in {1..N}).
link c(X,Y) >>> c(X,Y1) molecule field rate 10000.0 where ( Y1 in [Y-1,Y+1] ).
link c(X,Y) >>> c(X1,Y) molecule field rate 10000.0 where ( X1 in [X-1,X+1] ).

concentration 1 of pump into c(M,M) where (const size(N), M is N//2).
place AnyReaction into AnyCompartment.

final_steps 100000.
sample_steps 100.

out [molecule(c(X,Y),field),Delimiter] where (
    const size(N),
    inspect(compartment c(X,Y)),
    (Y=N -> Delimiter=end_of_line ; Delimiter=space)
).
```

Level 3: the Simulation Engine – Output (1)

Produces a textual result from out commands

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 4 3 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 7 54 7 3 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 12 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Level 3: the Simulation Engine – Output (2)

Produces a textual result from `out` commands

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 3 1 2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 2 4 7 9 11 6 0 0 0 0 0 0 0 0
0 0 0 0 0 0 5 6 24 29 30 9 1 0 0 0 0 0 0 0
0 0 0 0 0 1 3 8 31 85 56 19 4 0 0 0 0 0 0 0
0 0 0 0 0 1 3 20 83 166 73 21 9 1 1 0 0 0 0 0
0 0 0 0 0 1 2 9 58 90 56 14 6 1 0 0 0 0 0 0
0 0 0 0 0 0 0 2 19 27 26 10 3 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 5 7 5 1 2 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

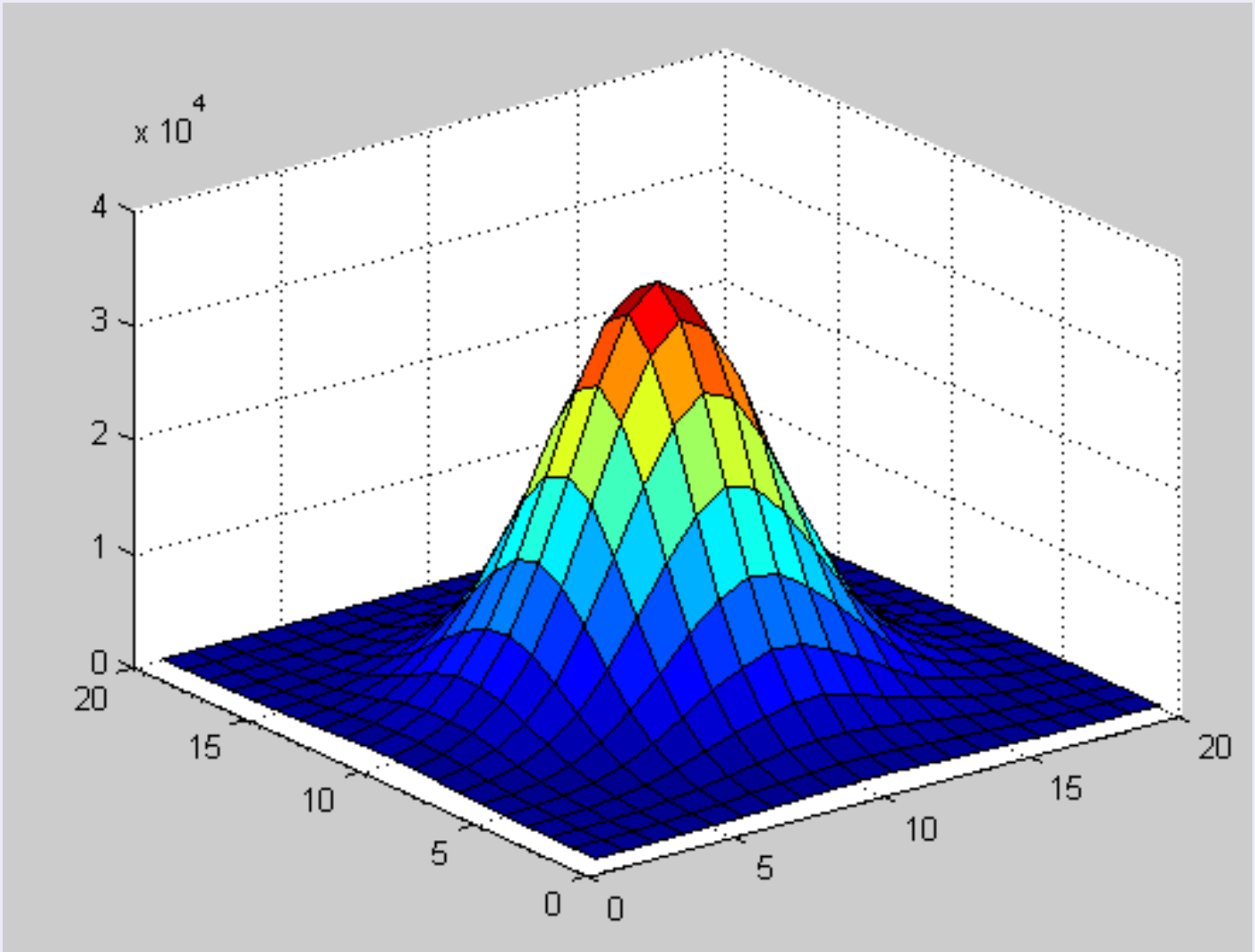
Level 3: the Simulation Engine – Output (3)

Produces a textual result from out commands

```
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 2 1 3 6 5 2 0 1 0 0 0 0 0 0
0 0 0 0 0 2 6 11 11 14 12 2 0 1 3 2 0 0 0 0
0 0 0 1 4 8 17 36 64 57 46 20 7 5 2 1 0 0 0 0
0 0 0 3 9 15 31 77 120 142 98 47 36 16 5 1 0 0 0 0
1 1 2 6 20 28 78 162 217 238 221 145 78 43 14 6 1 0 0 0
0 1 1 12 33 83 171 261 401 497 419 265 130 52 16 3 1 0 0 0
1 5 9 20 52 93 227 399 658 779 587 381 228 84 29 5 2 0 0 0
0 0 6 16 44 100 243 460 768 966 737 495 269 98 38 10 4 4 0 0
0 0 3 11 44 106 191 385 638 803 677 431 234 94 36 11 0 0 0 0
0 0 3 2 32 51 120 251 412 466 445 338 160 61 23 16 1 0 0 0
0 0 2 1 8 27 48 117 205 289 246 171 75 39 15 11 5 1 0 0
0 0 1 2 2 11 25 46 86 134 117 64 36 14 5 5 7 1 0 0
0 0 0 0 0 2 6 24 31 37 26 23 9 4 0 0 0 0 0 0
0 0 0 0 0 0 1 4 6 10 4 7 4 2 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 3 2 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Level 3: the Simulation Engine – Drawing Charts

Charting using any existing tool (Matlab, gnuplot,..)



Level 3: the Simulation Engine – Under the hood

Based on:

[1] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem., 81(25), 1977.
[2] M. A. Gibson and J. Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. J. Phys. Chem. A, 104(9), 2000.

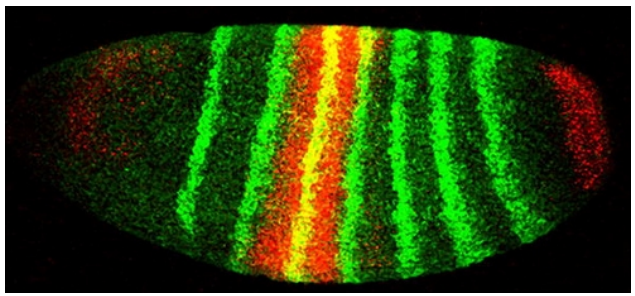
Main Elements

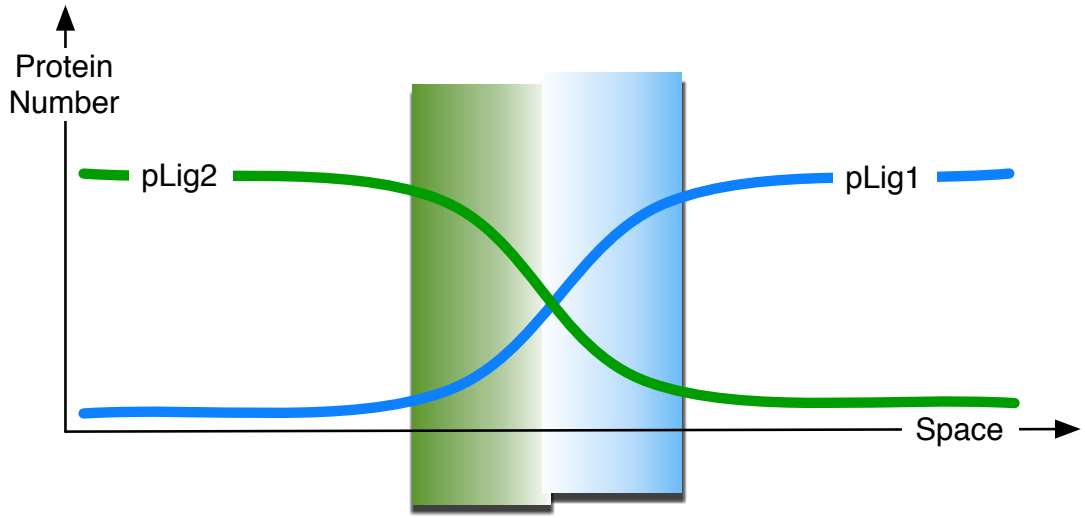
- An available chemical reaction in a compartment is picked up probabilistically based on its rate
  - selected in $O(logN)$ time, via a special binary search tree
- The transition modifies a small portion of data structures
- The transition duration is drawn with Poisson distribution
  - exactly modelling chemical dynamics according to [1]

Outline

Goal of the Model

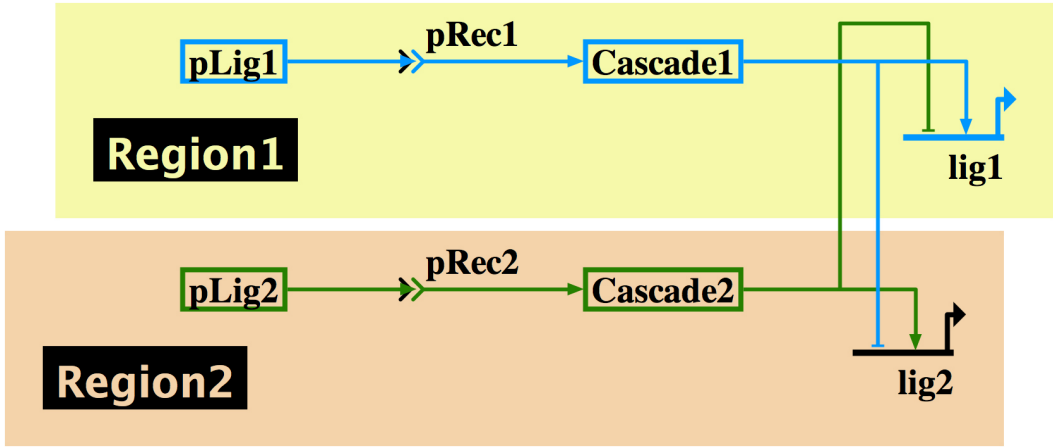
- To obtain a self-organized "stripes" pattern in a 2D scenario
  - regions with a different genetic expression (pLig1 and pLig2)
  - boundary regions with a gradient of activity of the expressed genes





Working Hypotheses of the Model

- The maternal products activate either pLig1 or pLig2
- Two regions where only one of the two pathways is active



- Two signalling pathways composed by
  1. Receptors → pRec1 & pRec2
  2. Ligands → pLig1 & pLig2
  3. Intracellular cascades that transmit the signal → cascade1 & cascade2
  4. Genes → lig1 & lig2

→ pLig1 and pLig2 inhibit each other

Model Specification: Intracellular Reactions

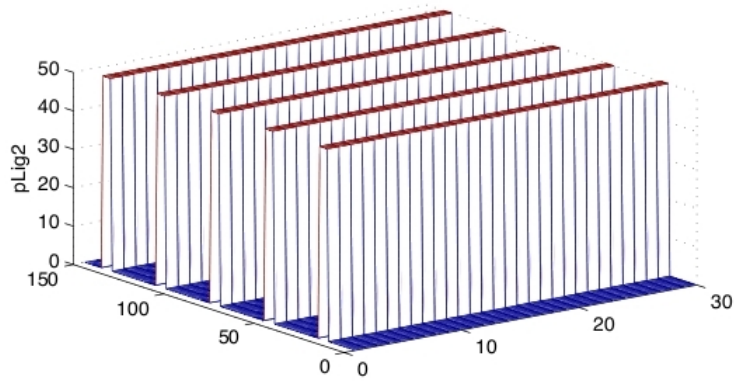
Reactions section

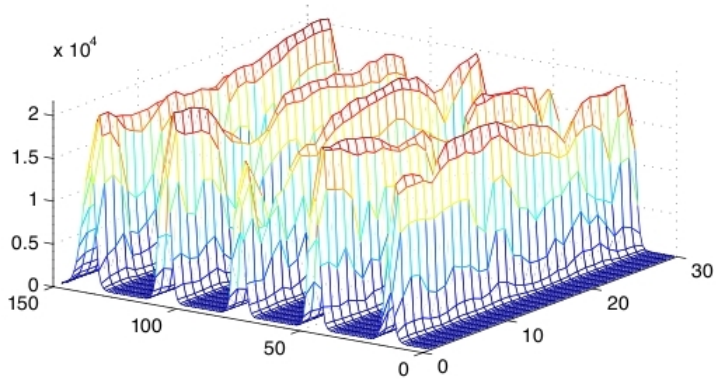
```
reaction r(pLig1Born)  : [mpLig] --> [pLig1,pLig1] rate 10.0.
reaction r(pLig2Born)  : [mpLig] --> [pLig2,pLig2] rate 10.0.

reaction r(pRec1Activation) : [pLig1,pRec1] --> [pLigRec1] rate 1000.0.
reaction r(pRec1DeActivation) :  [pLigRec1] --> [pLig1,pRec1] rate 100.0.
reaction r(pCasc1Phosphorylation) : [pLigRec1,casc1] --> [pLigRec1,casc1P] rate 1000.0.
reaction r(pCasc1DePhosphorylation) : [casc1P] --> [casc1] rate 1.0.
reaction r(gLig2Inhibition) : [casc1P,gLig2] --> [gLig2I] rate 100000.0.
reaction r(gLig2DeInhibition) : [gLig2I] --> [casc1P,gLig2] rate 0.02.
reaction r(gLig1Activation) : [casc1P,gLig1] --> [gLig1A] rate 10000.0.
reaction r(gLig1DeActivation) : [gLig1A] --> [casc1P,gLig1] rate 0.02.
reaction r(pLig1Synthesis) : [gLig1A] --> [pLig1,gLig1A] rate 10.0.
reaction r(pLig1Degradation) : [pLig1] --> [] rate 0.03.

reaction r(pRec2Activation) : [pLig2,pRec2] --> [pLigRec2] rate 1000.0.
reaction r(pRec2DeActivation) :  [pLigRec2] --> [pLig2,pRec2] rate 100.0.
reaction r(pCasc2Phosphorylation) : [pLigRec2,casc2] --> [pLigRec2,casc2P] rate 1000.0.
reaction r(pCasc2DePhosphorylation) : [casc2P] --> [casc2] rate 1.0.
reaction r(gLig1Inhibition) : [casc2P,gLig1] --> [gLig1I] rate 100000.0.
reaction r(gLig1DeInhibition) : [gLig1I] --> [casc2P,gLig1] rate 0.02.
reaction r(gLig2Activation) : [casc2P,gLig2] --> [gLig2A] rate 10000.0.
reaction r(gLig2DeActivation) : [gLig2A] --> [casc2P,gLig2] rate 0.02.
reaction r(pLig2Synthesis) : [gLig2A] --> [pLig2,gLig2A] rate 10.0.
reaction r(pLig2Degradation) : [pLig2] --> [] rate 0.03.

reaction r(moveLig1)  : [pLig1] --> [firing(pLig1)] rate 100.0.
reaction r(moveLig2)  : [pLig2] --> [firing(pLig2)] rate 100.0.
```

First Experiment – pre-imposed regionalisation

Initial configuration

- 4500 cells forming a 30x150 grid
- Few copies of the two ligands already regionalised at the beginning of zygotic expression (left)





Discussion

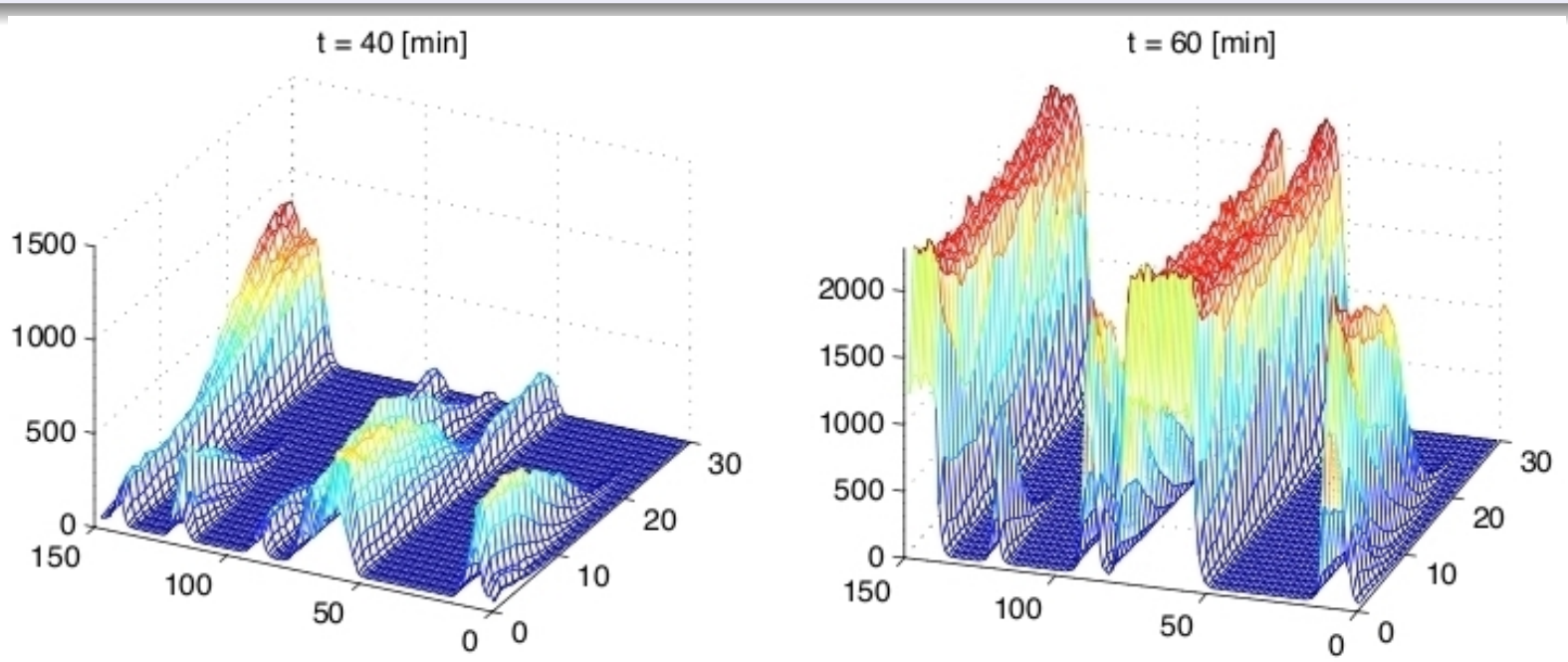
The initial pattern is strongly reinforced (right) thanks to:

- mutual inhibition + local reinforcement

Second Experiment – spontaneous regionalisation

Initial configuration

- 4500 cells forming a 30x150 grid
- same number of ligands uniformly placed



Discussion

- regions tend to form (left) and consolidate (right)
- dynamics is not very stable: some further mechanism is needed

Conclusions & Future Works

Agenda

- Re-engineer the tool towards a community release
- Support dynamic networks and mitotic division
- Applications in supporting biologists' understanding of morphogenesis mechanisms

A consideration..

- There is need for more advanced implementation techniques to tackle efficiency of simulations!

A Computational Framework for Modelling (and Simulating) Multicellular Biochemistry

Sara Montagna Mirko Viroli

{sara.montagna, mirko.viroli}@unibo.it

Alma Mater Studiorum—Università di Bologna a Cesena

IEEE Congress on Evolutionary Computation (CEC 2009)
Trondheim, Norwey, 18th–21st May 2009