Chapter 1
Interaction-Based Simulations for Integrative Spatial Systems Biology
Application to the Simulations of a Synthetic Multicellular Organism in MGS

Antoine Spicher, Olivier Michel, and Jean-Louis Giavitto

Abstract Systems biology aims at integrating processes at various time and spatial scales into a single and coherent formal description to allow analysis and computer simulation. In this context, we focus on rule-based modeling and its integration in the domain-specific language MGS. Through the notions of *topological collections* and *transformations*, MGS allows the modeling of biological processes at various levels of description. We validate our approach through the description of various models of a synthetic bacteria designed in the context of the iGEM competition, from a very simple biochemical description of the process to an individual-based model on a Delaunay graph topology. This approach is a first step into providing the requirements for the emerging field of *spatial systems biology* which integrates spatial properties into systems biology.

Key words: Interaction-Based Simulations, Domain-Specific Languages, Rule-Based Modeling, Integrative Spatial Systems Biology, Synthetic Biology.

1.1 Introduction

It is Fermi et al. (1955) who proposed that computers, instead of simply performing standard calculus, could be used to study and test a physical idea. This was the

Antoine Spicher · Olivier Michel
LACL - EA 4219 - Université de Paris 12 - Paris EST
61 avenue du Général de Gaulle, 94010 Créteil Cedex - France
e-mail: {antoine.spicher,olivier.michel}@univ-paris12.fr

Jean-Louis Giavitto
IBISC Lab. - FRE 3190 CNRS, Université d'Évry & Genopole
523 place des terrasses de l'agora, 91000 Évry - France
e-mail: giavitto@ibisc.univ-evry.fr

introduction, in 1955, of the idea of *numerical experiments*, also called *in silico* experiments by biologists.

This epistemological and sociological change had far reaching consequences, providing to systems biology a unique tool in the investigation of biological phenomena. Computer modeling and simulation give to the biologist an access to "experimental results" that cannot be provided by direct experiments because of practical, economical or ethical reasons. However, as biologists realize the limitations of informal, intuitive analysis of complex systems (McAdams & Shapiro 1995, von Dassow et al. 2000), the computer is no longer used only to perform a computation that cannot be done analytically or by hand: its is also used to check and compare theoretical models, to systematically investigate the consequences of an hypothesis, to explore the possible range of the parameters and to record, analyze, control and summarize some elements of the (possibly non-deterministic) behavior of a complex biological system.

Within biology, systems biology is a particularly demanding application domain since it requires to integrate several models coming from unrelated area of science like mechanics, chemistry, *etc.* The computer modeling and simulation of such systems require the coupling of several model fragments specifying deterministic or stochastic interactions between the system's entities to represent continuous or discrete evolution. For instance, the modeling of the growth of the meristem at a cellular level (Barbier de Reuille et al. 2006*b*) requires the coupling of molecular mechanisms (*e.g.* chemical reaction, diffusion, active transport), mechanical stresses, developmental changes and genetic regulation.

Computer science has developed (or appropriated) many languages and tools to help build models of real-world processes and to relate different models that operate on different levels of abstraction and various spatial and time scales. In this chapter, we advocate the use of a *rule-based framework* based on *spatial interactions* as a unifying framework for the concise and expressive simulation of a broad class of biological systems. We will address related issues such as: Can the same framework be used to model deterministic and stochastic systems? Do we need different frameworks for the expression of continuous and discrete systems? Could the same approach allow the natural and concise expression of various theoretical approaches (for the purpose of simulation)? An answer to such questions cannot be derived theoretically but convincing elements can be provided through paradigmatic examples. This chapter is then organized as follows.

Section 1.2 discusses some of the requirements of systems biology models, the growing role of agent-based models and the current focus put on the notion of *interaction*. We emphasize also the need to handle explicit spatial relationships.

Section 1.3 presents MGS, a rule-based, spatial interaction-oriented, experimental programming language dedicated to the simulation of a broad class of biological systems.

Section 1.4 introduces the running example we use to illustrate the versatility of the rule-based approach: a synthetic multicellular bacteria or SMB. This example comes from a project presented at the iGEM contest in synthetic biology. SMB combines diffusion, genetic regulation and signalisation in a population.

Section 1.5 illustrates the use of the MGS rule-based approach with the development of several models of the SMB. Each model focuses on a specific time scale using a dedicated theoretical framework. We show how the MGS approach, emphasizing the notion of *spatial interactions*, is able to express concisely in the same unified and uniform simulation framework, stochastic and deterministic models and discrete and continuous ones.

A short presentation of the perspectives and challenges opened by this work concludes this chapter.

1.2 Computer Modeling and Simulation in Integrative and Spatial Systems Biology

In this section we sketch several approaches in the modeling of biological systems. We propose to base a unifying simulation framework on the spatial organization of the interaction between the entities that compose the system. An experimental programming language based on this idea is proposed in the next section and illustrated by several examples in the second part of this chapter.

1.2.1 Dynamical Systems in Systems Biology

Biological processes are often modeled as *dynamical systems* (Smith 1999). At any point in time, a dynamical system is characterized by a set of *state variables*. The evolution of the state over time is specified through a *transition function* which determines the next state of the system (over some time increment) as a function of its previous state and, possibly, the values of external variables (input to the system). The evolution function can be generalized to an *evolution relation* to handle non-deterministic (*e.g.* stochastic) evolution.

Various mathematical framework with diverse properties can be considered to formalize a dynamical system. For instance, state variables may take values from a continuous or discrete domain. Likewise, time may advance continuously or in discrete steps. Some examples of dynamical systems characterized by different combinations of these features are listed in Table 1.1. Other combinations exist and are not listed: the disintegration of a radio-active atom is a continuous-time Markov process with discrete state for instance.

These various formalisms can be applied to the same system to capture different aspects of the system's evolution. For example, the same reaction-diffusion process (Turing 1952) in a tissue can be modeled in continuous space by partial differential equations (PDE) or in a discrete space by a system of coupled ordinary differential equations (ODE) where the state variables are the concentration of morphogens in each cell (Turing did both in his seminal paper). Reaction-diffusion pro-

cesses can be also modeled by iterated mapping, sometimes called "continuous automata", a variant of von Neumann's cellular automata (CA) (Von Neumann 1966) where a cell is described by real-valued local concentrations (Turk 1991). And totally discrete (space, time and state) models of reaction-diffusion have also been proposed, for instance in Greenberg & Hastings (1978).

Table 1.1 Some formalisms used to specify dynamical systems according to the discrete or continuous nature of time, space and state variables. The "space" row is explained in section 1.2.2.2.

C: continuous D: discrete	PDE	ODE	Iterated Mappings	Finite Automata
State	C	C	C	D
Time	C	C	D	D
Space	C	D	D	D

1.2.1.1 The Need of a Unifying Simulation Language

The previous example shows that a simulation workbench for integrative biology cannot support a unique theoretical framework. In addition, even confronted to the development of one specific simulation, the programmer must cope with the wide variety of biological entities (genes, proteines, membranes, cells, tissue, *etc.*). They cannot be described in a unique formalism and yet they must be placed in a single simulation framework. This is also the case in multiscale modeling where models of the same system at different scales can have fundamentally different characteristics (*e.g.* deterministic *vs.* stochastic).

These observations do not imply that only a general programming language can be used for the implementation of simulations in systems biology. As a matter of fact, the notion of dynamical system is a very general one but nevertheless, it may receive some specific support that motivates the development of *domain specific languages* (DSL). DSL offer, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain.

We believe that it is possible to provide abstractions and notations generic enough to encompass and unify the variety of formalisms needed in systems biology. Such DSL will support the expressive representation of various kind of states, time and evolution functions as well as the building of coupled heterogeneous models such as discrete/continuous or stochastic/deterministic dynamical models. The programmer will be able to express the various models in a concise and expressive way, making easier to debug, tune and evolve the simulations. Such DSL makes also possible to relate models through their implementation.

1.2.1.2 State and Evolution function in Systems Biology

From the previous presentation, it is obvious that a DSL dedicated to simulation in systems biology must support in one way or another the notions of state and evolution function. However, these two simple notions must be looked in a fresh way in the context of systems biology. Indeed, most of the systems considered in biology consist of *populations* of *interacting entities*. A good example is a biological cell modeled by a system of molecules that react and interact to form (other) molecules and molecular machines.

It is customary to abstract over these entities and use state variables to denote macroscopic observables or population level properties like a global concentration or a temperature[1]. It is assumed that as the population size increases, the behavior of the biological system is asymptotic to that of this state-variable model.

The terms *aggregate* or *mean-field* are sometimes used to qualify this approach. It allows a concise expression of the model and despite severe limitations, mean-field approximations have been standard methodology for modeling populations of interacting entities, especially for large and homogeneous populations. One reason is that no viable alternative existed before the widespread availability of inexpensive computing power.

However, aggregate models rely on two assumptions that must be seriously scrutinized in systems biology:

1. the state space can be described *a priori* and remains fixed;
2. the global evolution function can be defined explicitly.

We examine these two assumptions in the rest of this paragraph.

1.2.1.3 Dynamical Systems with a Dynamical Structure

Very often the state space of the considered biological process cannot be described *a priori*. The reason is that the structure of the biological system and therefore its description (by a set of state variables) may itself vary over time, as pointed out by Giavitto, Godin, Michel & Prusinkiewicz (2002). An example is given by the development of an embryo. Initially, the state of the system is described solely by the chemical state of the egg. After several divisions, the state of the embryo is given not only by the chemical state of each cell, but also by their spatial arrangement. The number of cells, their spatial organization and their interactions evolve constantly in the course of the development and is not handled by one static structure. It means that the phase space used to characterize the structure of the state of the system at each time step must be computed jointly with the running state of the system.

The dynamicity of the structure of a biological system has been repeatedly emphasized and several formalisms have been proposed to specify both the evolution of states and the evolution of the structure. Examples include: the concept of

[1] relying on a *mean-field* approach where the idea is to replace all interactions to any entity with an average interaction, reducing any multiple entities problem into an effective one-entity problem.

(hyper)-cycle introduced by Eigen and Schuster in the study of auto-catalytic networks (Eigen & Schuster 1979), the notion of *autopoietic systems* formulated by Maturana and Varela (Varela et al. 1974, Luisi 2003), the *variable structure system* theory developed in control (Itkis 1976), or the concept of *organization* introduced by Fontana and Buss to formalize and study the emergence of self-maintained functional structures in a range of chemical reactions (Fontana & Buss 1994).

We call such systems *dynamical systems with a dynamical structure*[2] or $(DS)^2$ in short (Giavitto & Michel 2002*c*, 2003, Giavitto 2003). Biological examples include the production of molecules and their dynamic association into multimolecular complexes (Fontana 1992) or the birth and death of cells with their mechanical constraints and signaling relations within a developing organism (morphogenesis).

1.2.1.4 Local Interactions

A consequence of a dynamical structure is that a global evolution function cannot be specified. As a matter of fact, if the set of variables that describe the system cannot be known in advance, it is impossible to specify a *global* evolution function (a dynamical structure is not mandatory to prevent the explicit definition of a global evolution function).

This does not mean that the (global) evolution function does not exist: it simply cannot be defined explicitly. This is the case when the individual (local) interactions between the system's entities are well characterized but the corresponding global evolution function cannot be deduced. The macroscopic (global) evolution of the system must be computed as the "integration" of all the various local and dynamic interactions between entities.

1.2.2 Individual-Based Models and their Simulations

Individual-based models (Lynch 2008), also called *agent-based models*, propose an alternative approach to mean-field approximation. Such models describe a system from the perspective of its constituent units and focus on the representation of the evolution of each individuals that appears in the system. As a consequence, they tackle more easily the enormous modeling difficulties raised by the dynamical structure of biological systems.

Individual-based approaches attract a renewed interest and become viable alternatives because of the increasing availability of inexpensive computing power. However, they have their own drawbacks. Their mathematical analysis appears to be at least as difficult as analysing aggregate variables models and the simulation remains the main tool to study the system's evolution and for reaching conclusions.

[2] Bailly & Longo (2006, pp 125-127) recognize the importance of this class of dynamical systems and call it "dynamicité auto-constituante" (which could be translated to "self-producing dynamicity"), a distinctive feature of living organisms

Thus, beyond aggregate models, a simulation language dedicated to systems biology must be able to implement individual-based models.

1.2.2.1 Multi-Agent Implementation

Multi-agent systems (MAS) (Woolridge & Wooldridge 2001) are often advocated as the tool of choice for the implementation of individual-based models (Spicher et al. 2009). A multi-agent system is a collection of autonomous decision-making entities called agents. Each agent individually assesses its situation and makes decisions on the basis of a set of rules.

It is easy to use an agent in a multi-agent system to represent the state of an entity part of the modeled system. The global state of the system is then the set of the state of each agent that composes the system.

However, *multi-agent systems provide no support for the notion of interaction*. Several entities are engaged simultaneously in an interaction while agents are supposed to evolve autonomously. Admittedly, in determining its evolution, an agent takes into account its neighbors. But it cannot take into account for example the evolution of its neighbors, which can be problematic.

A good illustration is given by a simple model of growth sometimes called the Eden model (Eden 1961). It has been used since the 1960's as a model for tumor growth. In this model, a space is partitioned in empty or occupied cells. At each step, occupied cells with an empty neighbor are selected, and the corresponding empty cell is made occupied. An exclusion principle prevents two occupied cells to invade the same empty cell.

This specification of the local evolution of the system defines the interaction between an occupied and an empty cell. It is difficult to turn this specification into a simple rule for *one* cell evolution because an empty cell can query its neighborhood to find if they are occupied cells but they cannot known which or even if an occupied cell will invade it. Conversely, when an occupied cell decides to invade an empty one, it cannot determine if another occupied cell makes the same decision at the same time.

1.2.2.2 The Spatial Structure of Interactions

Usually, only physically close entities interact because information exchange ultimately has a *local* character (*e.g.* transport of signaling molecules between neighboring cells). Thus the possible interactions of the entities in the system reflect the underlying physical space. The other way round, we can say that the spatial organization of the entities composing the system organizes also their interactions.

In Table 1.1, we have introduced an additional criterion, *space*, to categorize dynamical systems formalisms following the discrete or continuous setting used for the spatial organization of their entities. For instance, in a cellular automaton,

entities called “cells”, are organized in a regular lattice. In PDE, fields are localized in a continuous space and can extend over an entire subspace.

It is interesting to examine the case of aggregate models. For example, the aggregate model of a chemical reaction supposes that the chemical solution is well stirred and abstracts a population of molecules by a set of concentration variables. In this case, there is no need to record the position and the velocity of each molecules in the continuous underlying physical space: it is as if each molecule could interact with any other. The possibility of chemical reactions is then only constrained by the “compatibility” of the reactants and is better described by a discrete structure, the molecular interaction network. Even if this network represents functional constraints rather than constraints from the physical underlying space, it is obtained by “erasing” the localization of the molecules keeping only the possibility of interactions between different species.

Taking the previous discussions seriously pushes to make a switch from state and evolution function to individuals and interactions. In the next section we will see how topological notions used to describe neighborhood relationships can be used to support the description of interactions in a programming language.

1.3 The MGS Domain-Specific Programming Language

The previous section points out several difficulties raised by the computer modeling and the simulation of biological processes in the context of integrative and spatial systems biology: the necessity to accomodate a wide range of mathematical formalisms (from continuous to discrete and from deterministic to stochastic), the dynamical structure of the system and the specification of the dynamics through local, spatially organized interactions.

To face these difficulties and to ease the building of a simulation program, we advocate a language-based approach through a *Domain Specific Language* (DSL) dedicated to the simulation in systems biology. DSLs are programming languages for solving problems in a particular domain. To this end, they provide abstractions and notations for the domain at hand. They are more attractive for programming in the dedicated domain than general-purpose languages because of easier programming, systematic reuse, better productivity, reliability, maintainability, and flexibility. Moreover, DSLs are usually small, and more often declarative than imperative. Declarative programming focuses on *what* should be computed instead of *how* it must be done. Objects and constructions are close to the mathematical standards which enable an easier mathematical reasoning on programs. Thus, a declarative program is an executable specification not burdened by the implementation details and is close to the mathematical model.

In the rest of this section we present such a DSL : the MGS modeling language[3]. MGS allows a clear and concise specification of processes through spatial interac-

[3] The Web site of the project is `http://mgs.spatial-computing.org`

tions. In MGS, the state of a dynamical system is specified using an original and generic data structure: the *topological collection* (Giavitto & Michel 2002*a*). Topological collections are based on the topological relations between the interacting subparts of the system. Furthermore, the specification of the evolution law, trough local interactions, is simplified by the definition of *transformations*. Tranformations are functions defined by a set of rules. Topological collections and transformation are handled in a *declarative* style.

The notions of topological collection and transformation subsume several models of computation inspired by biological systems or used in their modeling and in their simulation like cellular automata, Lindenmayer systems (used in growth plant modeling) or P systems (advocated for the modeling of compartmentalized molecular interaction networks). The MGS approach is illustrated in section 1.5. The benefits of the MGS approach have been demonstrated through a number of complex applications in system biology, Cf. section 1.6.

1.3.1 Topological Collection

One of the key features of the MGS language is its ability to describe and manipulate *a collection of entities structured by a neighborhood relationship*. Such device, called a *topological collection*, is used to represent the state and the organization of a biological systems: the elements of the collection are the components of the system, and the topology of the collection sets the potential interactions (i.e., two elements in the collection may interact only if they are neighbor).

Intuitively a topological collection generalizes the notion of *field* widely used in physics: each collection is build on an underlying space by associating some value to each *position* in this space. Positions can be points, but also more generally lines, surfaces, volumes, *etc.* The value associated with a surface may represent a flux, the value associated with a volume may represent a concentration, *etc.*

Topological collections can also be thought as a generalization of the notion of array where the index of an element is replaced by a position in the underlying space (Giavitto & Michel 2002*a*). This view subsumes a large family of important data-structures used in simulations. For instance, a labeled graph is a special case of a topological collection where the positions are the nodes of the graph and the neighborhood relationships are given by the edges of the graph.

Technically, the formalization of topological collections relies on the notion of *chain complex* (Munkres 1984) defined in algebraic topology and has been thoroughly studied in previous work of the authors (Giavitto & Michel 2002*c*, Giavitto & Spicher 2008*b*).

Several neighborhood relationships are expressible in MGS. In the rest of this chapter we will mainly use *records*, *multiset*, *Group-based Fields* (GBF), and *Delaunay* collections. These collections represent several important families of interactions and we will show that they are homogeneously handled in MGS. In addition, MGS allows heterogeneous collections (the elements of a collection can have differ-

ent types) and the arbitrary nesting of collections (*i.e.*, an element of a collection can itself be a collection). These features greatly facilitate the development of models and their simulations.

1.3.2 Transformation

Topological collections represent an adequate medium to specify the interactions between the elements of a biological system. In MGS, the specification of a *transformation T*:

`trans` *T* `= {  ...  σ =>` $f(\sigma,\ldots)$`;  ...  }`

corresponds to the definition of a set of rules, where the left-hand side σ is a pattern, matching for a subcollection, and the right-hand side $f(\sigma,\ldots)$ is an expression that evaluates a new subcollection that will be inserted in place of the matched one. The notion of subcollection depends on the neighborhood relationships of the collection: a subcollection is a connected subset of a collection and two elements are connected if they are neighbors.

Intuitively, a rule represents a possible (local) evolution of a (sub)system. The pattern in the left-hand side of the rule represents a potential configuration and the expression in the right-hand side computes the local evolution of this configuration.

A very simple transformation is given by:

```
trans simpleT = { 0 => 1; }
```

This transformation is composed of a single rule which replaces the value `0` in the collection by the value `1`. There are two important points to note.

First, this transformation may be applied to any kind of collections. Such a transformation is called *polytypic* (Jansson & Jeuring 1997). Polytypic transformations encapsulate an abstract process that can be reused in a variety of situations. For example, MGS is expressive enough to allow the definition of a generic diffusion process that can be used on any kind of collections (Giavitto & Spicher 2008*b*).

Secondly, if the transformation `simpleT` defines the replacement of `0` by `1`, it does not specify which `0`'s must be replaced. If there are several occurrences of `0` in the collection, do we have to replace all of them, some of them or just one of them? In the two latter cases, how are the occurrences chosen? These choices are under the control of a *rule application strategy*. The application of the transformation *T* on a topological collection *e* using a strategy *St* is written

T`[strategy =` *St*`](`*e*`)`

In the current implementation of MGS, all available strategies are built-in (but the functional composition of the transformations allows a certain flexibility for specific requirements). In the following, we will use two of them: the *Gillespie strategy* based on the stochastic simulation algorithm proposed by Gillespie to simulate chemical reactions (Gillespie 1977) (see section 1.5.3) and the *maximal-parallel*

strategy widely used in the context of L-systems (Lindenmayer 1968*a*) and P systems (Păun 2001). In the maximal-parallel strategy, which is the default strategy, a maximal set of non-intersecting occurrences of the pattern are simultaneously replaced by the right-hand side of the rule. When several such sets exist, one of them is non-deterministically chosen.

1.3.3 Two Models of Diffusion

We illustrate the notion of transformation with the simulation of a paradigmatic diffusion process. Diffusion is defined in a continuous setting in one dimension by the following equation

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2}$$

where D is the diffusion coefficient, u is the concentration of the diffusing substance and x is the position. Below, we describe two different approaches to simulate the diffusion of a chemical on a one dimensional rod.

1.3.3.1 The Numerical Resolution of the Continuous Model

This example shows MGS' ability to handle a continuous model. By their nature, computer simulations operate in discrete time. Models initially formulated in terms of continuous time and space must therefore be discretized. Using a simple finite difference method, the previous equation is discretized as

$$u(i,t+dt) = u(i,t) + h\sum_j (u(j,t) - u(i,t))$$

where $u(i,t)$ represents the concentration at time t of the i^{th} element of the discretized rod, and j ranges over the neighbor of i. Parameter h depends on the discretization and on the diffusion constant D. This computation can be programmed in MGS by the following transformation:

```
trans diffuse[h] = {
  u => u + h * (neighborsfold(+,0,u)
                - u * neighborsize(u))
}
```

where `u` is a pattern variable that matches any element in a collection, the expression `neighborsfold(`*op*`,`*e*`,u)` uses operation *op* to combine the values of the neighbors of `u` starting from the initial value *e* and `neighborsize(u)` returns the number of neighbors of `u`. An additional parameter is provided between brackets after the name of the transformation and corresponds to the parameter *h*.

It is straightforward to extend this process to a surface or a volume instead of a 1D rod. More elaborate discretization schemes are also handled similarly: for example, we give in the annex of this chapter the MGS program corresponding to the implementation of the Range-Kutta methods.

1.3.3.2 The Discrete Stochastic Evolution of a Diffusing Particle

Now, we want to take the same system but we focus on the level of the molecules. The rod is discretized as a sequence of small boxes, indexed by a natural integer, each containing zero or many molecules. At each time step, a molecule can choose to stay in the same box, or to jump to a neighboring box, with the same probability p (whose value depends on the time and space discretization). The state of a molecule is the index of the box where it resides. The entire state of the system is then represented as a *multiset* of indices.

A multiset is a generalization of a set (Banâtre et al. 2006): the same element can appear multiple times in a multiset. In a multiset, each element is neighbor of any others. Thus, a multiset is a good idealization of a well stirred "chemical soup" (Giavitto et al. 2004). In our example, if there are m molecules in the box numbered n, then there is m occurrences of the integer n in the multiset.

The evolution of the system can then be specified as a transformation with three rules:

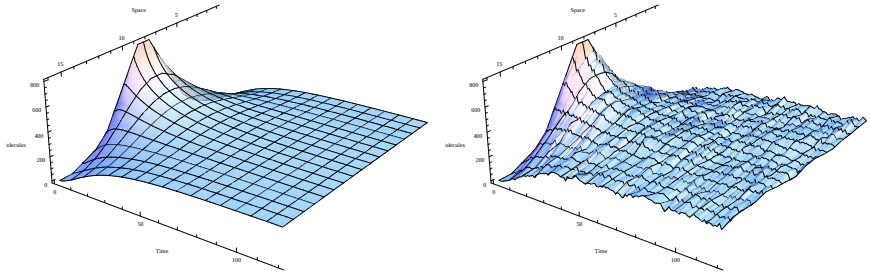
```
trans diffuse[p] = {
    q ={ P = (1 - 2*p) }=> q
    q ={ P = p         }=> q + 1
    q ={ P = p         }=> q - 1
}
```

The arrow construction ={...}=> is used to specify the specific parameters of a rule. Here we give a value to the parameter `P` used in the probabilistic application of the rule. In this strategy, a matched pattern is replaced by the right-hand side of the rule only with a probability `P`. Additional rules (not shown here) are provided to deal with boundary conditions.

Fig. 1.1 illustrates the iteration of the continuous and stochastic transformations. In the initial state, all particles are randomly distributed in the middle third of the rod.

1.4 A Synthetic Multicellular Bacterium

In the forthcoming sections, we propose to illustrate the expressiveness brought by the MGS language for the modeling, at various spatial and time scales, of the same biological process: a *synthetic multicellular bacterium* (SMB) built during the 2007 iGEM competition (Brown 2007) by the french team in Paris (Bikard et al. 2007).



(a) Continuous diffusion (b) Discrete diffusion

Fig. 1.1 Evolution of a chemical diffusing in a 1D rod, modeled as a continuous process (a) or as a discrete stochastic one (b). Intuitively, the left figure is the limit of the right figure when the number of boxes in the rod and the number of particles grow to infinity.

We start by a short presentation of synthetic biology, the iGEM competition and then we describe the SMB project of the Paris team.

1.4.1 Synthetic Biology

Synthetic biology is an emergent field which proposes an engineering point of view on biology. It aims at building new biological systems by assembling standard low-level components called *BioBricks* (Knight 2006). These components, designed in the projects presented for the iGEM contest, are described and stored in an ontology hosted by the MIT[4]. They are pieces of DNA used to build biological functions (as for example a logical gate), and integrable within existing genomes. For example, a brick activating the production of a chemical species in the presence of a sufficient concentration of molecules of types *A* and *B* can be interpreted as a function calculating the conjunction of the chemical signals associated with the species *A* and *B*.

The basic principles of construction of the biological components, establishing the *biosynthetic* methodology, was elaborated at the MIT at the turn of the 21st century. They rely on classical engineering strategies: *standardization*, *decoupling* and *abstraction* (Endy 2005). The purpose of *standardization* is twofold: to ensure compatibility between the bricks and to allow the development of generic and normalized building protocols (*i.e.* functioning for all bricks), economically accessible and easily implementable. *Decoupling* is a strategy that separates complicated problems into simpler ones. For instance, the separation of the various functions of a synthetic system allows the modularization of the system, the reuse of its parts, the

[4] The BioBricks are available in the *Registry of Standard Biological Parts* at the following url `http://partsregistry.org/Main_Page`.

independent evolution of each of them, *etc.* The separation of the phases of design and implementation reduce and eliminates the dependence between the design of a gene regulatory network and the effective building of a strand of DNA. *Etc.* Finally, an *abstraction hierarchy* supports the engineering of integrated genetic systems by hiding information and managing complexity through relevant levels of expression: from DNA nucleotides to parts, devices and (complete biological) systems. Abstraction levels limit the exchange of information across levels and allow individuals to work at any level without regard for the details that define other levels.

1.4.2 The international Genetically Engineered Machine (iGEM) Competition

iGEM is a competition launched by the MIT in 2003. More than 110 teams coming from all around the world participated in the 2009 issue. The competition is aimed at undergraduate students that are given the opportunity to manipulate complex molecular biology processes made simple by the synthetic biology principles. During a three months time period, students mentored by post-graduate students and researchers, design, model and assemble BioBricks to produce *new biological functions* integrated into living systems. At the end of the competition, all teams gather at the MIT in the first week-end of November during the *Jamboree* where their projects are being evaluated.

In 2007 a french team supervised by A. Lindner and S. Bottani participated in the competition and was ranked first in the "foundational research" category for their *Synthetic Multicellular Bacterium* project. MGS was used to produce most of the simulations needed to validate the design (one simulation was done in MATLAB). In section 1.5 we present several simulations that are inspired or extend the initial SMB simulations.

1.4.3 Objectives of the SMB Project

The objective of the SMB project is the design of a synthetic multicellular bacterium. This organism was thought as a tool that would allow the expression of a lethal or dangerous transgenic gene in the *Escherichia coli* bacterium without disturbing the development of its biomass. The main difficulty was to install a mechanism of irreversible bacterial differentiation which makes possible to express the transgene only in a part of the population unable to reproduce. The two lines, *germinal* (not differentiated) and *somatic* (differentiated and unable to reproduce) are interdependent and then constitute a multicellular organization (hence the name "*multicellular* bacterium"). In order to ensure that the ratio between the two populations makes it possible for the system to grow, the sterile somatic cells are designed to

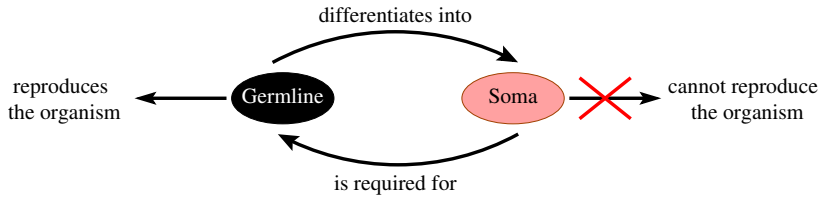


Fig. 1.2 The SMB is composed of two cell types: germ cells (G) and somatic (S) cells. G cells are able to live by producing two different types of cells: G cells and S cells. S cells are derived from G cells by an irreversible differentiation step, exhibiting a new function required for the survival of the G cells. S cells cannot reproduce. This dependency between G and S cells defines the organism.

provide to the germinal cells a molecule essential to their reproduction: DAP (diaminopimelate). Fig. 1.2 sketches the general principle of the project[5].

The design of this organization asked for the development of two distinct biological functionalities, one for the cellular differentiation and the other for the feeding of DAP to the germinal cells. The study of this system was at the same time theoretical and practical. Although the biological implementation of the system could not be entirely carried out by lack of time, the students at iGEM Paris provided experimental evidences and theoretical proofs that the SMB organism was viable.

1.4.4 The Paris Team Proposal

To implement this functionality into the E.coli bacterium, the Paris team has proposed an original construction. The gene regulatory networks of the proposal is described in Fig. 1.3. Two functions are described: a *feeding device* based on the production of DAP molecules (light gray) and a *differentiation device* based on a classical Cre/LOX recombination scheme (dotted box).

In the germline G, there is a natural expression of ftsK. This gene is essential for replication. The protein product of gene dapA is DAP. This protein diffuses in the environment and is rapidly degraded. However, in the germline, the dapA gene is not active since it lacks a promoter to initiate its transcription and G is auxotrophic in DAP.

The promoter dapAp is sensitive to DAP concentration. Located before the gene Cre, it allows to adjust the production of Cre to the presence of DAP in the environment. The production of Cre initiates the recombination/differentiation process.

After recombination, the genomic reassembly leads, by the excision of the parts between the two LOX recombination sites, to the cell of type S and a plasmid that is rapidly degraded. In the feeding device S, dapA is under the control of its constitutive promoter and can be expressed. The synthesized DAP diffuses in the en-

[5] Additional informations are available at (Bikard et al. 2007).

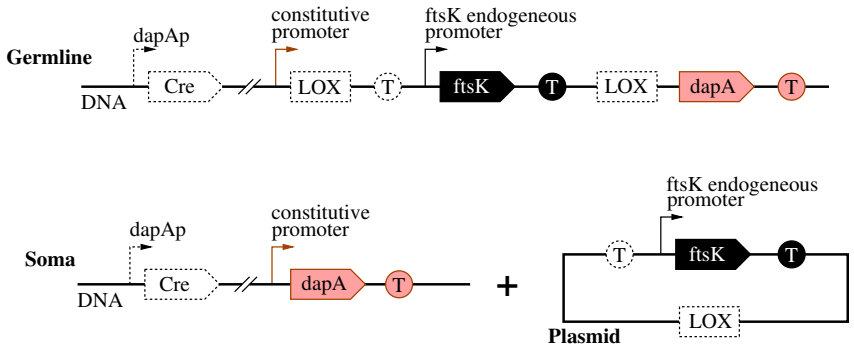


Fig. 1.3 Gene regulatory networks of the germinal and somatic cells describing the feeding device (light gray) and differentiation device (dotted box). Cre, dapA and ftsK are genes, LOX is a recombination site and T are terminators.

vironment allowing to reach G cells. Lacking the ftsK gene, S cells are sterile and eventually die.

1.5 Modeling in MGS

In this section we illustrate the expressive power of MGS through four examples derived from the SMB. These four examples have been chosen in order to illustrate the MGS concepts on individual-based models as well as aggregated models, and on spatialized as well as non-spatialized models, Cf. Table 1.2.

Table 1.2 Aggregated models vs. individual-based models and spatialized vs. non-spatialized models in the SMB simulation examples.

	aggregated	individual-based
non-spatialized	ODE (sect. 1.5.1)	stochastic simulation *à la* Gillespie (sect. 1.5.3)
spatialized	discrete diffusion of DAP (sect. 1.5.2)	cell-cell dynamical interaction (sect. 1.5.4)

1.5.1 Solving Differential Equations

This first modeling of SMB is a kind of proof of concept based on the study of a differential equations system. We propose here a rule-based expression of this model with two simple resolution schemes: the Euler and Runge-Kutta methods.

1.5.1.1 The SMB Proof of Concept

The very design of SMB is based on the composition of a feeding device together with a differentiation device. We wonder here whether this architecture could reach homeostasis, no matter how these devices are implemented. So a minimal model is required to give such a proof of concept of the design.

To answer this fundamental question, the Paris team proposed a theoretical study of the population dynamics based on a classical differential equations model. Let $[G]$, $[S]$ and $[D]$ denote the concentration of germinal cells, somatic cells and DAP molecules in a well-mixed solution. Their dynamics are captured by the three following equations:

$$\frac{d[G]}{dt} = \alpha_1 \frac{[D]^n}{[D]^n + k^n}[G] - \alpha_2[G] - \alpha_3[G] \tag{1.1}$$

$$\frac{d[S]}{dt} = \alpha_2[G] - \alpha_4[S] \tag{1.2}$$

$$\frac{d[D]}{dt} = \alpha_5[S] - \alpha_6[D] \tag{1.3}$$

They give the time variation of each concentration as functions of $[G]$, $[S]$ and $[D]$. Parameter α_1 denotes the growth rate of germ cells, parameter α_2 denotes the differentiation rate, parameter α_3 denotes the death rate of germ cells, parameter α_4 denotes the death rate of somatic cells, parameter α_5 denotes the production rate of DAP by the somatic population, and parameter α_6 denotes the degradation rate of DAP. In this model, the differentiation device is parametrized by α_2 and the feeding device is captured by parameters α_5 for the DAP production and α_1 that is weighted by a Michaelis-Menten function representing the dependence of germinal cells growth to the DAP concentration.

1.5.1.2 Analysis of the ODE model

In general such models based on differential equations are not easily investigated. The parameters are often numerous and qualitative analyses are difficult. In our case, parameters α_5 and α_6 can be dropped assuming that the DAP concentration is stabilized (*i.e.* when $[D]$ remains constant and equation 1.3 vanishes). This simplification of the model allows to stress out two main populations behaviors. Indeed it reveals a non trivial fixed point $([G]_0, [S]_0)$ that is unstable:

- for greater values of cell concentrations, an exponential growth is observed,
- for lower values of cell concentrations, both populations collapse to reach the second and trivial fixed point $(0,0)$.

But is this result relevant? In other words, is the DAP stabilization assumption realistic? Should the production of DAP fluctuate, the previous sketch does not give any information on the viability of the SMB. In the following sections, we propose to focus on this question relying on different characterizations of the dynamics using numerical simulations.

1.5.1.3 A Numerical Solution of Differential Equations

By their nature, simulations operate in discrete time. Models initially formulated in terms of continuous time must therefore be discretized. Strategies for discretizing time in a manner leading to efficient simulations have extensively been studied. Here we use as an example a straighforward and very simple approach, the Euler method. This method particularly fits well the simulation of problems of the form:

$$\frac{d\mathbf{X}(t)}{dt} = f(\mathbf{X}(t)) \qquad \mathbf{X}(0) = X_0$$

where $\mathbf{X}(t)$ is a vector of values representing the state of the system at a given time t, and X_0 is the initial state. The function f computes the variation of each coordinate of $\mathbf{X}$ at a given time t. As far as our problem is concerned, one has the state $\mathbf{X} = ([G],[S],[D])$ and function f corresponds to the three equations (1.1), (1.2) and (1.3).

The Euler method computes a sequence of vectors $\mathbf{X}_n$ where $\mathbf{X}_0 = X_0$ at the initial time and the generic term is given by the first two terms of the Taylor expansion:

$$\mathbf{X}_{n+1} = \mathbf{X}_n + \Delta t\, f(\mathbf{X}_n)$$

where Δt denotes the simulation time step.

We start the MGS expression of this computation by representing the state of the system in terms of topological collection. We use here a *record*. A record is one of the simplest collection consisting of two or more values so that each component (called a *field* or *member* of the record) can be accessed through a symbolic name. Each value in the record is "isolated" and has no neighbor. Hence, there is no direct interaction between the elements of a record. Elements of a record are given between braces.

The record used here has three members describing the concentrations $[G],[S]$ and $[D]$ with a value of type `float` (a real number):

```
record State = { G:float, S:float, D:float }
```

The variation of each concentration can be computed from such a state. The following function implements this procedure according to equations (1.1), (1.2) and (1.3):

```
fun Variation[a1,a2,a3,a4,a5,a6,k,n](X) = {
  G = (X.D^n / (X.D^n + k^n) *a1 - a2 - a3)*X.G,
  S = a2*X.G - a4*X.S,
  D = a5*X.S - a6*X.D
}
```

Parameters ai, k and n are given between bracket. Parameters between brackets are optional arguments. Note that the function Variation returns a State. It allows collections X and Variation(X) to be of the same type, and then to share the same set of positions (here fields G, S and D). This property eases the computation. For example, while the concentration $[G]$ is obtained at position G of collection X (by the expression X.G), its variation $\frac{d[G]}{dt}$ is at the same position G of collection Variation(X) (corresponding to the expression Variation(X).G).

Finally, one step of the Euler method can be expressed by a transformation to be applied on a collection of type State:

```
trans Euler[dt=Δt,f] = {
  x => let dx = f(self).(^x) in x + dt * dx
}
```

In this transformation, the unique rule specifies how each element x of the collection has to be updated by computing its variation dx. This variation is taken at ^x (i.e. the position of x) of the collection and is computed by the function f (a parameter of the transformation) applied on self. The identifier self always refers to the collection which the transformation is applied to. In our example, the actual value of f will be the previous function Variation. The whole trajectory is obtained by iterating the application of transformation Euler on an initial condition.

The reader is invited to note that transformation Euler is fully independent from the specification of State and Variation, and can be used as a generic implementation of the Euler method in many different contexts. Moreover, whereas the Euler method is sufficient for the simulations described below, we would like to underline that other integration methods can also be straightforwardly implemented in MGS. The implementation of the Runge-Kutta method is elaborated in appendix 1.6 of this chapter.

1.5.1.4 Interpretation of the Simulations' Results

Numerical approaches suffer from the lack of knowledge regarding the values of parameters. Hence, we cannot rely on any quantitative information on the system. Nevertheless, experience and classical examples give us sufficient informations to determine a range of possible parameters. For the sake of the simplicity, we arbitrarily set them to the intervals given in Table (1.3).

Table 1.3 Intervals of the parameters for the ODE-based model.

Parameter	Range
α_1	[0,2]
α_2	[0,1]
α_3	[0,1]
α_4	$\alpha_4 = \alpha_3$

Parameter	Range
α_5	[0,1000]
α_6	[0,1]
n	2
h	100

Our objective was to observe all the possible behaviors of the system for different settings of parameters (chosen in the parameters space defined by Table 1.3) and starting from a common initial state.

The protocol of our study has consisted in running 10000 simulations of the model. Each simulation has consisted in computing the Euler trajectory of the system over 11000 iterations with a time step equal to 0.01 (that is, 110 arbitrary units of simulation time) starting from an initial state where only germinal cells are present with a very high concentration of DAP. In each run, parameters were randomly chosen according to the intervals given in Table (1.3).

```
Euler[dt=0.001,f=Variation]
     ({ G = 100, S = 0, DAP = 10000 })
```

Results are given in Fig. 1.4. Only three clearly distinguished behaviors are observed and coincide with the dynamics provided by the qualitative analysis:

1. population collapse (see Fig. 1.4.(a)),
2. exponential growth of the population (see Fig. 1.4.(b)), and
3. the unstable fixed point (see Fig. 1.4.(c)).

In all behaviors, the system starts by consuming DAP molecules to replicate. Once DAP concentration falls below a certain level, differentiated cells start to appear and initiate the production of DAP.

In order to understand what characteristics prevent an exponential growth, the simulations have been classified according to the three behaviors, and their distributions have been analyzed (see Fig. 1.4.(d)-1.4.(h)). Each histogram shows the behavior of the system following one parameter and irrespectively of the other ones. Population collapse occurs when germinal replication rate is low (either because of a small growth rate α_1 or because of a high death rate α_3, see Fig. 1.4.(d) and 1.4.(f)) or when the differentiation rate is too low (see Fig. 1.4.(e)). The last two Fig. 1.4.(g)-1.4.(h) show that the system is not perturbed by the behavior of DAP production or degradation. This explains why no additional behaviors are observed compared to the qualitative analysis. In fact, the dotted lightgray curves of Fig. 1.4.(a)-1.4.(c) (that correspond to the ratio $\frac{[D]}{[G]+[S]}$) show that the normalized DAP concentration remains constant after a transient phase; the assumption $\frac{d}{dt}\left(\frac{[D]}{[G]+[S]}\right) = 0$ seems appropriate for a qualitative analysis.

The conclusion of this study is twofold:

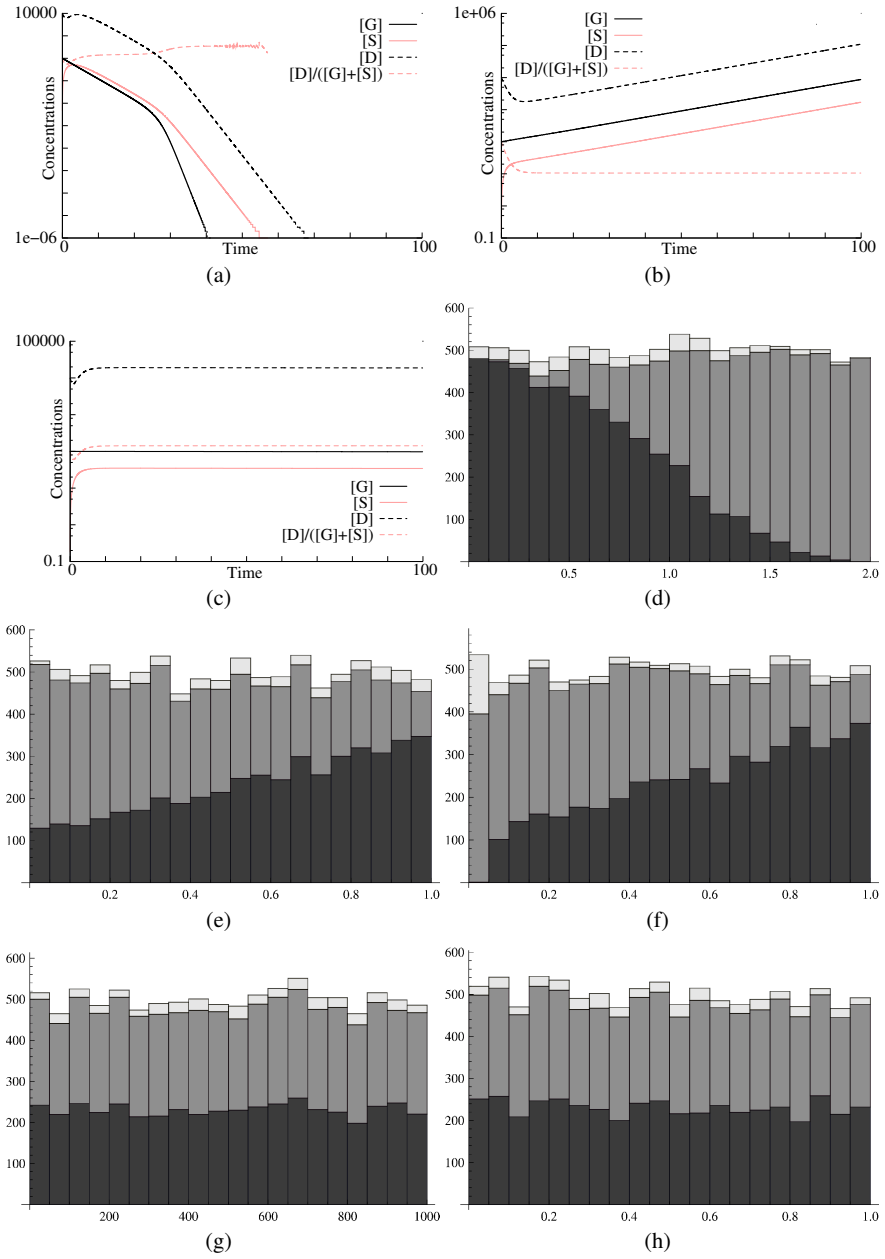


Fig. 1.4 Results of simulations of the ODE-based model. Fig. 1.4.(a)-1.4.(c) illustrate the three observed behaviors (resp. population collapse, exponential growth and unclassified behavior). Fig. 1.4.(d)-1.4.(h) give on stacked histograms the distribution of the 10000 simulation runs for each parameter (resp. α_1, α_2, α_3, α_5, α_6) with populations collapse in medium gray, exponential growth in dark grey, and unclassified behavior in light gray.

1. the choice of DAP as the main molecule to design the feeding device is good if the auxotroph germ line is robust and grows well in presence of DAP;
2. the differentiation device is required to be efficient (a reversible differentiation should be prohibited).

We aimed here at illustrating how MGS can be used as a prototyping tool for providing quick results and orienting further investigations. This preliminary work could be improved by taking into account more realistic parameters ranges provided by the litterature. The use of data analysis techniques could also give a better knowledge (*e.g.* parameters correlations) on the results of the simulations.

1.5.2 Cellular Automata

In this second modeling, we focus on the effects on the SMB due to the spatial organization of the SMB. We propose to design a cellular automaton and to implement it in MGS.

1.5.2.1 The Spatial Organization of the SMB

The ODE-based model proposed in section 1.5.1 considers the SMB as a molecular solution of three different species uniformly distributed in space. In reality, the system consists of two populations of cells that will be organized in space. Such an organization may induce heterogeneity in the cells distribution leading to some spatial artefacts not taken into account by the ODE model (Durrett & Levin 1994). Some interesting spatial self-organizations could even be observed: for instance, one can easily imagine that the SMB collapses at some locations while it grows exponentially at others making some patterns appear at the population scale (Shnerb et al. 2000). As a consequence, one has to investigate whether space matters or not in the SMB development.

1.5.2.2 A Discrete Spatial Framework

Different formalisms allow to take space into account. A first direction consists in extending the ODE of section 1.5.1 by considering the spatial distribution of concentrations (*i.e.* $[G]$, $[S]$ and $[D]$ would depend on time but also on space). This extension would introduce in the formula the use of two additional terms to deal with the spatial diffusion of concentration of bacteria and DAP molecules. These modifications make the original ODE system become a PDE system, and increase the paramater space. The associated phase space becomes more difficult to study.

Cellular automata (CA) and multi-agents systems (MAS) are another class of formalisms that explicitly consider spatial organization. Both rely on a individual-based point of view. We focus here on a CA approach.

A cellular automaton is a regular lattice of places called "cells", where each cell is characterized by a state taken from a finite set. The global evolution of the CA consists in applying synchronously, on each cell, a local evolution function that computes the new state of the cell as a function of its current state and of the states of the cells in its neighborhood.

The Paris team proposed a CA to study the relation between DAP diffusion and differentiation. Their model is based on states encoded as real numbers to represent the concentration of DAP in each CA cell, and the use of non-deterministic rules using random number generators. On the contrary, we propose a totally deterministic CA with discrete states and very simple rules. More precisely, we consider a superposition of two CA: the one deals with the DAP diffusions process while the other takes into account the differentiation of the cells.

DAP Diffusion CA

Contrarily to the simulations of diffusion given in section 1.3.3, we aim at specifying a phenomenological diffusion in a CA, that is the propagation of an information (*e.g.* "contains some DAP") from a source CA cell to its neighborhood. This behavior corresponds to a classical propagation rule (Wolfram 1986) where a cell becomes activated if one of its neighbors is activated. In order to limit the radius of the propagation, the following rule may be used:

$$x_{t+1} = \max(0, y_t^1 - 1, \ldots, y_t^n - 1) \tag{1.4}$$

where x denotes a cell of the CA, x_t its state at time t and $y^1, \ldots, y^n$ the neighbors of x. Here states are encoded by integers that are gradually decremented from the source: 0 means no activation and $n > 0$ means that the activation propagates with a radius n around the cell. Some evaporation may be introduced to deal with the removal of the source. A source maintains its state to a specific integer value denoted by $N_{\mathscr{R}}$. Fig. 1.5.(a)-1.5.(c) show the discrete diffusion around an isolated source for $N_{\mathscr{R}} = 5$ on an hexagonal grid.

Differentiation CA

This CA focuses on the bacterial layer. Each cell of the CA represents a part of the whole population. Under some conditions on the DAP level, a CA cell progressively goes from a majority of germinal bacteria to a majority of somatic bacteria. We abstract the ratio between the two populations in a CA cell by an integer between 0 and $N_{\mathscr{P}}$: 0 means that only somatic bacteria are present and $N_{\mathscr{P}}$ only germinal bacteria are present.

The dynamics of a CA cell is as follows: if there is enough DAP, the number of germinal bacteria becomes maximal. Otherwise, this number decreases at each time step. Finally, if no germinal bacterium remains, the ratio is locked to 0. Equa-

tion (1.5) summarizes this behavior:

$$u_{t+1} = \begin{cases} N_{\mathscr{P}} & \text{if there is } \textit{enough} \text{ DAP} \\ 0 & \text{if } u_t = 0 \\ u_t - 1 & \text{otherwise} \end{cases} \tag{1.5}$$

where u denotes a CA cell and u_t its state at time t.

Coupling Both CA

Equations (1.4) and (1.5) are combined to define the final CA. Let $c_t = (u_t, x_t)$ denotes the state of a CA cell c a time t. The local evolution function is given by:

$$c_{t+1} = \begin{cases} (0, \quad N_{\mathscr{R}}) & \text{if } x_t = 0 \\ (u_{t+1}, \max(0, x_{t+1} - N_{\mathscr{C}})) & \text{otherwise} \end{cases} \tag{1.6}$$

where x_{t+1} and u_{t+1} are given by equations (1.4) and (1.5), and $N_{\mathscr{C}}$ represents the DAP consumption of germinal bacteria. Finally, we consider that there is not *enough* DAP when $x_{t+1} - N_{\mathscr{C}}$ is negative and no DAP source is available in its neighborhood.

1.5.2.3 MGS Expression of a Cellular Automaton

MGS allows an easy specification of CA. We use the *Group-based field* topological collections (GBF) to represent regular and uniform lattice, as used in cellular automata or for the numerical solutions of PDEs. The neighborhood relationships of a GBF are described in terms of a mathematical group, the group of elementary displacements in the lattice (Giavitto & Michel 2001, Giavitto, Michel & Cohen 2002). The corresponding space can be visualized as a graph, the Cayley graphs of the presentation of the group (Giavitto & Michel 2002*b*). This abstract approach enables the easy specification and a uniform handling of a large family of circular and screwed, bounded or unbounded grids in any dimension.

For the purpose of this example, each GBF position is labelled by an MGS record of type {`x:int, u:int`} representing the state c^t. Each GBF position has 6 neighbors achieving an hexagonal grid.

The dynamics of equation (1.6) is implemented as follows:

```
trans SMB_CA = {
  c / c.u == 0 => { x = N_R,          u = 0      }
  c / NoPrs(c) => { x = Diff(c)-N_C,  u = N_P    }
  c / c.u == 1 => { x = N_R,          u = 0      }
  c            => { x = 0,            u = c.u-1  }
}
```

where `NoPrs(c)` computes whether there is not enough DAP on cell `c` and `Diff(c)` computes the diffusion on cell `c` with respect to equation (1.4). Note that the order of the rules matters: for instance, the matching of a cell by the third rule implies that it cannot be matched by the first two ones. This transformation is applied using the standard maximal parallel strategy of MGS.

1.5.2.4 Interpretation of the Simulations' Results

Fig. 1.5.(d)-1.5.(f) show how differentiation appears in a population of germinal cells. As the CA transition function is deterministic, symmetry is broken in the initial state (otherwise all cells would exhibit the same behavior): we have chosen to start the simulation with cells of the form $\{$ `x = 0, u = 1+random(`$N_{\mathscr{R}}$`)` $\}$, that is to say germinal cells with a ratio uniformly chosen in $[1, N_{\mathscr{R}}]$, and no DAP.

No symmetrical pattern appears during the simulation, whatever the parameters $N_{\mathscr{R}}$, $N_{\mathscr{P}}$ and $N_{\mathscr{C}}$ are. The distribution of the differentiation cells follows the ratio distribution chosen at the initial state. An equivalent behavior is observed when the symmetry is broken by randomly initializing the field `x`. The uniformity of the

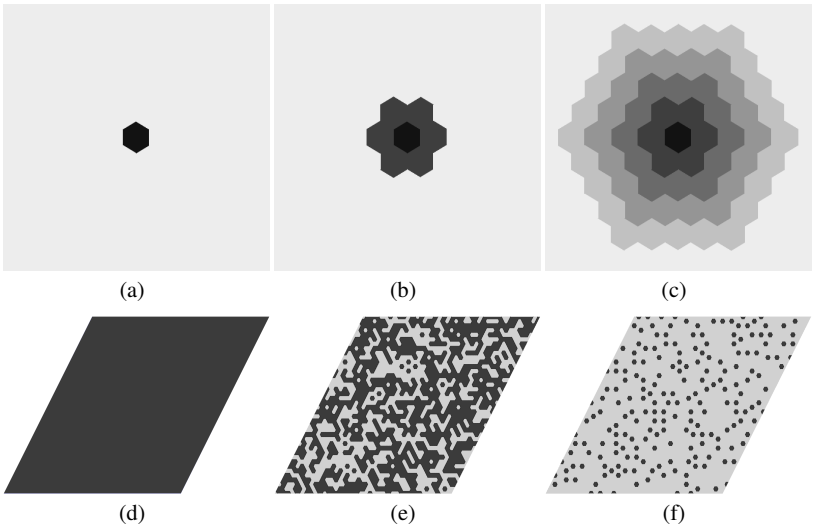


Fig. 1.5 Results of simulations of the CA model. Top line shows the propagation of DAP around an isolated source with radius 5: from left to right, initial state, state after 1 iteration, steady state. A light gray cell means no DAP, the gray scale represents the concentration of DAP and the black cell is the source. Bottom line shows the evolution of the CA defined by equation 1.6, on a 40x40 hexagonal grid only filled by germinal cells with a randomly chosen ratio: from left to right, initial state, state after 13 iterations, state at fixed point. Germinal cells figure in dark gray and somatic cells in light gray.

dynamics supports the assumption of a well-mixed solution used in section 1.5.1 and confirms the previous result.

We have shown with this model how a rule-based programming style fits well the specification of CA. No more than 10 lines are required to describe it in MGS. Moreover, thanks to the polytypic feature of MGS, the specification of the topology is totally decoupled from the definition of the dynamics; transformation `SMB_AC` could be applied on any kind of topological collection, and more especially on any kind of grids and neighborhoods (like square grids with von Neuman or Moore neighborhoods, toric grids, *etc.*). More specialized CA tools are often *ad hoc* and do not exhibit so much flexibility and genericity in the expression of models.

1.5.3 Stochastic Simulations

The two previous approaches provide results at the level of the synthetic device. In this section, we study the construction of these devices in terms of biological parts and synthetic construction as described in section 1.4.1. More specifically, we propose a stochastic model of the SMB at the level of one bacterium together with its simulations using the exact *stochastic simulation algorithm* defined by Gillespie (1977).

1.5.3.1 Robustness Analysis of the SMB Design

The characterization of a synthetic device depends on its implementation. We aim at checking if the behavior of the Paris team construction respects the main objective of the SMB. More precisely, we focus on the noise sensitivity and the relation between parts parameters (like the rate of DNA Cre/LOX recombining) and the devices parameters (such as the differentiation rate).

A common way of modeling gene regulation is to consider the regulatory network as a set of biochemical reactions. The set of chemical interactions induced by the Paris team construction (see Fig. 1.3) are abstracted by the following reactions:

$$\mathrm{Cre} \xrightarrow{C_0} . \tag{1.7}$$

$$\mathrm{DAP} \xrightarrow{C_1} . \tag{1.8}$$

$$\mathrm{D_{Cre}} + \mathrm{DAP} \underset{C_{-2}}{\overset{C_2}{\rightleftharpoons}} \mathrm{D^*_{Cre}} \tag{1.9}$$

$$\mathrm{D_{Cre}} \xrightarrow{C_3} \mathrm{D_{Cre}} + \mathrm{Cre} \tag{1.10}$$

$$\mathrm{D^*_{Cre}} \xrightarrow{C_4} \mathrm{D^*_{Cre}} + \mathrm{Cre} \tag{1.11}$$

$$\mathrm{D_G} + \mathrm{Cre} \xrightarrow{2C_5} \mathrm{D^*_G} \tag{1.12}$$

$$\mathrm{D^*_G} + \mathrm{Cre} \xrightarrow{C_5} \mathrm{D_S} \tag{1.13}$$

$$\mathrm{D_S} \xrightarrow{C_6} \mathrm{D_S} + \mathrm{DAP} \tag{1.14}$$

$$\mathrm{DAP} \underset{C_{im}}{\overset{C_{ex}}{\rightleftharpoons}} \mathrm{DAP}_{ex} \tag{1.15}$$

These reactions involve two kinds of chemical species: the DAP and Cre molecules, and the DNA constructions of Fig. 1.3 abstracted by:

- D_{Cre}, D^*_{Cre}: differentiation-free part of the construction composed of promoter dapAp and the coding region for Cre. The two symbols represent, respectively, the activated (no DAP repression on dapAp) and inhibited (DAP binds dapAp) state of the promoter.
- D_G, D^*_G, D_S: part of the DNA modified by the Cre/LOX recombination mechanism, D_G before and D_S after recombination. D^*_G corresponds to an intermediate state where only one LOX site is bound by Cre.

Reactions (1.7) and (1.8) describe the natural degradation of molecules Cre and DAP. Reactions (1.9-1.11) express the behavior of the promoter dapAp: inhibition/activation by DAP and production of Cre (reactions (1.10) and (1.11) differ in their reaction constants: $C_4 \ll C_3$). Reactions (1.12) and (1.13) specify the 2 steps of a Cre/LOX recombination: $D_G \rightarrow D^*_G \rightarrow D_S$. The regulation induced by D_G (expression of the gene ftsK) is not considered in this model. On the contrary, the behavior of D_S is specified by equation (1.14), corresponding to a constitutive production of DAP. The last reaction (1.15) expresses importation and exportation of DAP from the extracellular environment, where DAP_{ex} denotes the external occurrences of DAP molecules.

1.5.3.2 Stochastic Modeling for Sensitivity to Noise Analysis

The Paris iGEM team has chosen to investigate this kind of molecular model using a differential equation approach based on the mass action law. Thanks to this study, they provided a set of optimized parameters for an exponential growth of the SMB. Nevertheless, such results may be biased since the differential approach (1) relies on a global homogeneous assumption and (2) does not take noise into account. Since the number of molecules involved in gene regulation is in general very low, a stochastic approach may provide complementary result on noise sensitivity.

A usual abstraction in the simulation of biochemical systems consists in considering the system (here the bacterium) as an homogeneous chemical solution where the reactions of the model are taking place. Gillespie has proposed in (Gillespie 1977) an algorithm for producing the trajectories of such a chemical system by computing the *next reaction* and the *elapsed time* since last reaction occurred. Let μ be a chemical reaction, the probability that μ takes place during an infinitesimal time step is proportional to:

- c_μ, the *stochastic reaction constant*[6] of reaction μ;
- h_μ, the number of distinct molecular combinations that can activate reaction μ;
- $d\tau$, the length of the time interval.

Gillespie proved that the probability $P(\tau,\mu)d\tau$ that the next reaction will be of type μ and will occur in the time interval $(t+\tau, t+\tau+d\tau)$ is:

[6] Evaluating the stochastic constants is one of the key issues in stochastic simulations of biochemical reactions. The interested reader should refer to (De Cock et al. 2003, Zhang et al. 2003) for the description of two experiences in that field.

$$P(\tau,\mu)d\tau = a_\mu e^{-a_0\tau}d\tau$$

where $a_\mu = h_\mu c_\mu$ is called the *propensity* of reaction μ, and $a_0 = \sum_\nu a_\nu$ is the combined propensity of all reactions.

This probability leads to the first straightforward Gillespie's algorithm called the *first reaction method*. It consists in choosing an elapsed time τ for each reaction μ according to the probability $P(\tau,\mu)$. The reaction with the lowest elapsed time is selected and applied on the system making its state evolve. A new probability distribution is then computed for this new state and the process is iterated.

1.5.3.3 Gillespie-based Simulations in MGS

Here, we consider the bacterium as a well mixed chemical solution. It can be represented by a multiset, that is a topological collection, where any element may interact with all the others (Fisher et al. 2000). The simulation of "real" chemical reactions requires a strategy for multiset rules application in accordance with the reactions kinetics. The MGS language provides such a strategy based on Gillespie's algorithm. We propose to use this strategy for simulating the previous set of chemical reactions.

As said above, the state of the bacterium is represented by a multiset of values. The considered molecules are abstractly represented using MGS *symbols* denoted by back-quoted identifiers. For example, the MGS symbol `` `Cre `` corresponds to one molecule of Cre. Thus, each chemical reaction is translated into a transformation rule (or two if the reaction is reversible) characterized by an arrow parameter `C` representing the stochastic constant of the reaction. For example, the reversible reaction (1.9) can be straightforwardly translated to the two following MGS rules:

```
`dCrA, `DAP ={ C = C2  }=> `dCrI          and
`dCrI       ={ C = C-2 }=> `dCrA, `DAP
```

Consequently, the whole dynamics is captured by the following set of rules in transformation `SMB_STO`:

```
trans SMB_STO[DAPEx] = {
  `Cre         ={ C = C0          }=> .
  `DAP         ={ C = C1          }=> .
  `dCrA, `DAP  ={ C = C2          }=> `dCrI
  `dCrI        ={ C = C-2         }=> `dCrA, `DAP
  `dCrA        ={ C = C3          }=> `dCrA, `Cre
  `dCrI        ={ C = C4          }=> `dCrI, `Cre
  `dG1, `Cre   ={ C = 2*C5        }=> `dG2
  `dG2, `Cre   ={ C = C5          }=> `dS
  `dS          ={ C = C6          }=> `dS, `DAP
  `DAP         ={ C = Cex         }=> (DAPEx++; .)
  .            ={ A = DAPEx*Cim   }=> (DAPEx--; `DAP)
}
```

In the last two rules, the external DAP_{ex} molecules are specified by the counter `DAPEx` given as an optional parameter. This counter is incremented (resp. decremented) when a DAP molecule is imported (resp. exported).

The last rule of transformation `SMB_STO` explicitly computes the propensity `A` instead of using the usual parameter `C`. This feature allows a fine control of the Gillespie application strategy when required.

1.5.3.4 Interpretation of the Simulations' Results

A simulation is run by calling the transformation `SMB_STO` using Gillespie's strategy:

```
SMB_STO[strategy='Gillespie, DAPEx=1000]
       ('dCrA::'dG1::():bag)
```

The initial state is specified by two molecules, namely D_{Cre} and D_G, added (with the insertion operator `::`) to an empty multiset (denoted by `():bag` in the MGS syntax). This state represents a germinal cell. External DAP is specified by initializing the counter `DAPEx` to 1 000 molecules.

Top line of Fig. 1.6 gives two different runs of the simulation corresponding to the evolution of Cre, DAP and DAP_{ex} populations over 100 arbitrary units of time (*AUT*).

Fig. 1.6.(a) shows the classical behavior of a germinal cell: the DAP_{ex} is imported from the outside until no DAP remains in the system (this process takes 50 arbitrary units of time). During this first part of the simulation, the expression of Cre is repressed by the over representation of DAP. After that, the repression decreases during 20 *AUT* until some Cre molecules are produced. At time 80 *AUT*, DAP molecules appear which means that the differentiation occurred. On the contrary, Fig. 1.6.(b) shows the evolution of a germinal cell when DAP_{ex} remains constant (*i.e.* expressions `DAPEx++` and `DAPEx--` are removed from the specification of `SMB_STO`). DAP molecules are exchanged between the cell and its environment until an equilibrium is reached. Whereas any differentiation occurs during this simulation, the germinal cell will differentiate since parameter C_4 is not null.

We propose to use the stochastic model to evaluate the differentiation rate of the SMB. More specifically, we focus on the mean simulation time required for a germinal cell to differentiate while DAP_{ex} is constant. Results are given on the bottom line of Fig. 1.6. The protocol of this experiment consists in running 1 000 simulations for each value of $DAP_{ex} \in [0,30]$ and starting with the same initial state. Each simulation stops when the differentiation occurs (*i.e.* when `'dS` appears in the collection). Mean times and associated standard deviation are plotted on Fig. 1.6.(c). Surprisingly both quantities seem to behave *linearly* with DAP_{ex}.

One has to pay attention to the fact that such simulations are costly in computing time. Fig. 1.6.(d) gives the mean computation time of the simulation, showing that it increases more than linearly with the value of DAP_{ex}. Actually, Gillespie's algorithm, in its original definition, only allows a small numbers of molecules.

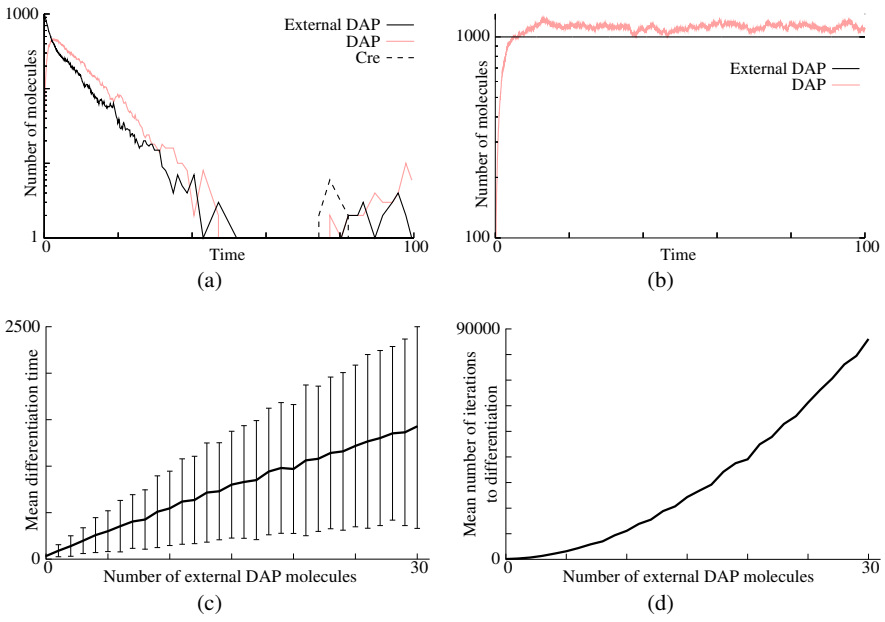


Fig. 1.6 Results of simulations of the stochastic-based model. On top, two examples of stochastic simulations when external DAP (in solid black) remains constant (Fig. 1.6.(b)) or not (Fig. 1.6.(a)). Internal DAP is drawn in light gray and Cre in dotted line. Fig. 1.6.(c) shows the mean simulation time to differentiation with a constant external DAP concentration for different values of that concentration. Vertical bars correspond to the standard deviation. Fig. 1.6.(d) shows the mean computation time to differentiation.

As a conclusion, one can establish that the differentiation rate is easily related to the quantity of DAP released in the environment by somatic cells. Such a result is meaningful as it relates quantities of different scales: the population and cellular scale of the differentiation and the genetic and molecular scale of DAP concentration.

Our stochastic simulations exhibit low concentrations of chemicals within a cell (*e.g.* there are less than 10 molecules of Cre after the differentiation occurs). This result questions the pertinence of ODE models such as the one proposed by the Paris team (Bikard et al. 2007).

1.5.4 Integrative Modeling

So far, we have considered classical ways of modeling and simulating a biological process at a given level of description. In this last model, we aim at simulating the

entire population of bacteria with an explicit representation of cells in a 2D space[7]. In addition, we want to integrate in the model physical and biological behaviors. Our proposition is based on the specification of cell-cell dynamical interactions and the computation of the neighborhood of the cells using an implicit Delaunay triangulation.

1.5.4.1 Description of the Model

In our proposition, bacteria are represented by circles localized in the 2D Euclidean space, with a radius depending on their size. Bacteria push away each other and consequently change their position in space and their immediate neighbohood. Thus, this neighborhood is required to be dynamically computed according to the spatial coordinates of their associated circles. This approach has already been successfully used in systems biology for the modeling of cell population (Gibson et al. 2006, Barbier de Reuille et al. 2006*a*).

The modeling of SMB is organized into two coupled models: a mechanical model and a biological model. A cell is encoded by a record which includes the position, the radius, the local DAP concentration, the differentiation state (germinal or somatic), *etc.* Cells are elements of a Delaunay topological collection. This kind of collection computes implicitly and transparently the Delaunay triangulation of a set of entities embedded in $\mathbb{R}^n$ (Aurenhammer 1991).

Mechanical Model

The mechanical model consists of a mass/spring system. Bacteria are considered as ponctual masses localized at the center of their associated circle; the presence of a spring between two masses depends on the neighborhood computed by the Delaunay triangulation. The mechanical effects of the growth of the bacteria is captured by the elongation of the springs rest lengths. Thus, each cell computes its acceleration by summing all mechnical forces induced by its incident springs, and consequently moves in space. This is done by the transformation `Meca`[8]. `Meca` sums the forces applied on each cell using a `neighorsfold` expression. Then, the `Euler` transformation (see section 1.5.1) is used twice to integrate during a time step $\Delta_1 t$ acceleration into velocity and velocity into new positions.

[7] The third dimension is not considered as the SMB is supposed to grow in the plane of a Petri dish for example.

[8] The whole MGS program of the simulation is available at `http://mgs.spatial-computing.org/integrative_biology.tgz`.

Biological Model

The cellular model is an extension of the CA given in section 1.5.2. The discrete DAP diffusion is replaced by the classical continuous model given in section 1.3.3, and a stochastic differentiation is used instead of the notion of populations ratio. New rules are added to deal with cellular growth, division and death: in presence of DAP, G cells grow by increasing their radius. When the G cell radius reaches a threshold, the cell divides. S cells keep on growing then die when another threshold is reached. The corresponding transformation is called `Cell`[8] and computes the cellular evolution during a period $\Delta_2 t$.

1.5.4.2 Integration of the Two Models

As classical functions, transformations can be arbitrarily composed. This is the key to the coupling of the two models. The iteration of a function can be specified by the MGS option `iter`. It allows to deal with different time scales: assuming that the mechanical process is faster than the cellular process (*i.e.* $\Delta_2 t > \Delta_1 t$), the whole model is captured by the following evolution function:

$$\texttt{fun SMB\_DEL(state) =}$$
$$\texttt{Cell[dt=}\Delta_2 t\texttt{](Meca[dt=}\Delta_1 t\texttt{, iter=}\frac{\Delta_2 t}{\Delta_1 t}\texttt{](state))}$$

where option `dt` corresponds to the time step used in transformations `Meca` and `Cell`. Here transformation `Meca` is applied $\frac{\Delta_2 t}{\Delta_1 t}$ times for only one application of `Cell`.

1.5.4.3 Interpretation of the Simulations' Results

The protocol of the proposed simulation consists in iterating 10 000 times the function `SMB_DEL` starting from a small initial population of 25 germinal cells with a deficit in DAP. Screenshots of the simulation are shown on Fig. 1.7. The vizualization of the evolution exhibits two interesting phenomena.

On small population size (*i.e.* at the beginning of the simulation), the ratio of the two cell lines fluctuates. Fig. 1.7.(b) presents a state of the population where most of the cells are germinal. On Fig. 1.7.(c), S cells predominate. The oscillations are due to delays between configurations where G cells are well fed (many S cells are present) and configurations of DAP starvation (not enough S cells are present). The fluctuations decrease as the population size increases and the ratio globally stabilizes as predicted by the ODE-based model (see Fig. 1.4). Indeed, the spatial distribution blurs the synchronization between cells.

The population tends to spatially organize in the following way: uniformly distributed small clusters of G cells surrounded by somatic ones. Clusters remain small-sized because when their size increase (by G cell divisions), the interior cells lack

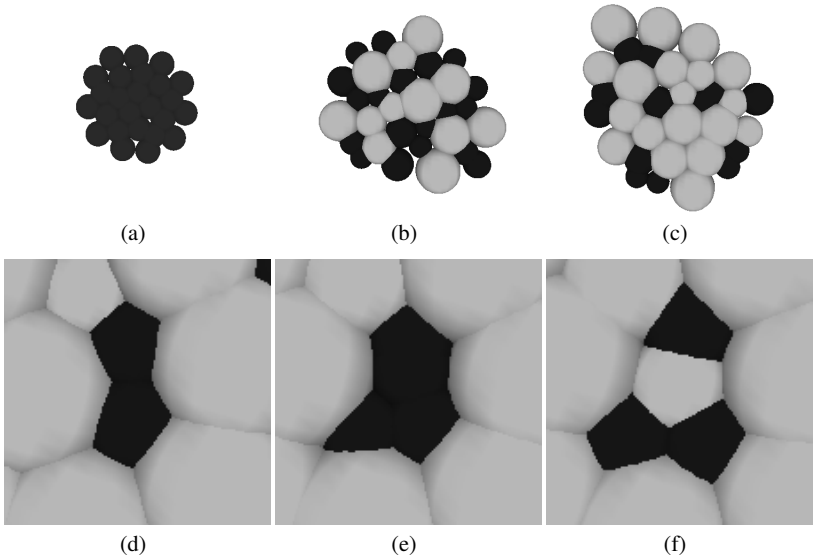


Fig. 1.7 Results of the integrative model. Germinal cells are in dark gray and somatic cells in light gray. Fig. 1.7.(a)-1.7.(c) correspond to an initial population and its evolutions at time 43*AUT* and 62*AUT*. Fig. 1.7.(d)-1.7.(f) focus on a cluster of germinal cells surounded by somatic cells. See text for explanation.

DAP and differentiate. This dynamics is reminiscent of the behavior exhibited by the CA model of section 1.5.2.

1.6 Related Work, Conclusions and Perspectives

In this short conclusion section, we present related work, close to our approach, used for the modeling and simulation of biological systems. Then, we conclude by assessing the importance of using a single and coherent domain-specific language for the modeling, at various spatial scales, of biosystems.

1.6.1 Related work

In this chapter, we have illustrated the use of MGS, an experimental programming language that investigates the use of topological collections and transformations in the simulation of complex biological systems. Based on the notion of spatial interaction, MGS provides a unified simulation framework encompassing dis-

crete/continuous and deterministic/stochastic formalisms. Even though MGS is a prototype in constant evolution, MGS' concepts have been validated on numerous applications : the crawling of the sperm cell of *Ascaris suum* (Spicher & Michel 2005), a simplified version of neurulation (Spicher & Michel 2007), the growth of the meristem at a cellular level (Barbier de Reuille et al. 2006*b*), molecular self-assembly (Giavitto & Spicher 2008*a*), the modeling of the paradigmatic phage lambda genetic switch (Michel et al. 2008), *etc.*

As a side-effect, changing the topology of a collection makes it possible for MGS to emulate some well-known computational models. Transformation on multiset is reminiscent of multiset-rewriting (Banatre & LeMetayer 1993). Nesting multiset lead to P systems (Păun 2001) a new distributed parallel computing model based on the notion of a membrane structure. P systems are advocated for the modeling of compartmentalized molecular interaction networks. Lindenmayer systems (Lindenmayer 1968*b*), which loosely correspond to transformations on sequences or string rewriting, have long and successfully been used in the modeling of $(DS)^2$ in the domain of plant growth. And, as shown in section 1.5.2, it is straightforward to express cellular automata in MGS.

1.6.2 Conclusions and Perspectives

Work in systems biology generally puts a considerable emphasis on the end result, the model of a complex biological system, and neglects so far the problem of *building* this model. The construction of a model and its use, *e.g.* for simulation, is a difficult task, and it often requires the combination of many formalisms and complementary approaches. We want also to stress the importance of dynamical structures in biological systems. This kind of dynamical systems is very challenging to model and simulate. New programming concepts must be developed to ease their modeling and simulation.

Using MGS' topological collections and transformations allowed us to model our problem in the same formal framework: a mean-field approach using ordinary differential equations for a quick proof of concept of the synthetic construction (section 1.5.1), a discrete spatial model using CAs on various topologies allowing a finer analysis of the relations between diffusion and differentiation (section 1.5.2), a robustness analysis on noise sensitivity on the SMB parts (Cre/LOX recombination) and devices (recombination rate) parameters using a stochastic modeling (section 1.5.3). The last model integrates physical and biological constraints in a 2D model, making it possible to analyze the effects of the spatial distribution on the various possible configurations (section 1.5.4).

The perspectives opened by this work are numerous. At the language level, the study of the topological collections concepts must continue with a finer study of transformation kinds. A lesson learned from the use of MGS by biologists is the needs of user-friendly interfaces and of graphical tools for the analysis of the simulation's results. Another direction of developments is the coupling of MGS with

various databases and repositories to retrieve parameters or to store and reuse fragments of models.

The development of MGS is part of a "grand challenge" aimed at the development of a methodological and technological framework that, once established, will enable the sharing of models, the analysis and the derivation of predictive hypothesis from them.

Acknowledgements

The authors would like to thank the reviewers for their valuable comments on a first version of this chapter.

We gratefully acknowledge all the people that contributed to make the first french participation in iGEM in 2007 a success: the students, D. Bikard, F. Caffin, N. Chiaruttini, T. Clozel, D. Guegan, T. Landrain, D. Puyraimond, A. Rizk, E. Shotar, G. Vieira, the instructors, F. Delaplace, S. Bottani, A. Jaramillo, A. Lindner, V. Schächter; the advisors, F. Le Fevre, M. Suarez, S. Smidtas, A. Spicher and P. Tortosa.

Further acknowledgments are also due to J. Cohen, B. Calvez, F. Thonnerieux, C. Kodrnja and F. Letierce that have contributed in various ways to the MGS project.

This research is supported in part by the the University of Évry, the University of Paris-Est, the CNRS, GENOPOLE-Évry, the Institute for Complex Systems in Paris-Ile de France, the ANR white project AutoChem and the french working group GDR GPL/LTP.

References

Aurenhammer, F. (1991), 'Voronoi diagrams–a survey of a fundamental geometric data structure', *ACM Comput. Surv.* **23**(3), 345–405.

Bailly, F. & Longo, G. (2006), *Mathmatiques et sciences de la nature*, Hermann.

Banâtre, J.-P., Fradet, P. & Radenac, Y. (2006), 'Generalised multisets for chemical programming', *Mathematical Structures in Computer Science* **16**(4), 557–580.

Banatre, J.-P. & LeMetayer, D. (1993), 'Programming by multiset transformation', *Comm. of the ACM* **36**(1), 98.

Barbier de Reuille, P., Bohn-Courseau, I., Ljung, K., Morin, H., Carraro, N., Godin, C. & Traas, J. (2006*a*), 'Computer simulations reveal novel properties of the cell-cell signaling network at the shoot apex in arabidopsis', *PNAS* **103**(5), 1627–1632.

Barbier de Reuille, P., Bohn-Courseau, I., Ljung, K., Morin, H., Carraro, N., Godin, C. & Traas, J. (2006*b*), 'Computer simulations reveal properties of the cell-cell signaling network at the shoot apex in Arabidopsis', *PNAS* **103**(5), 1627–1632.

Bikard, D., Caffin, F., Chiaruttini, N., Clozel, T., Guegan, D., Landrain, T., Puyraimond, D., Rizk, A., Shotar, E. & Vieira, G. (2007), 'The SMB: Synthetic Multicellular Bacterium (iGEM'07 Paris team web site)', `http://parts.mit.edu/igem07/index.php/Paris` visited in May 2009.

Brown, J. (2007), 'The iGEM competition: building with biology', *Synthetic Biology, IET* **1**(1.2), 3–6.

De Cock, K., Zhang, X., Bugallo, M. F. & Djuric, P. M. (2003), Stochastic simulation and parameter estimation of first order chemical reactions, *in* '12th European Signal Processing Conference (EUSIPCO-2004)'.

Durrett, R. & Levin, S. (1994), 'The importance of being discrete (and spatial)', *Theoretical Population Biology* **46**(3), 363–394.

Eden, M. (1961), A two-dimensional growth process, *in* 'Proceedings of Fourth Berkeley Symposium on Mathematics, Statistics, and Probability', Vol. 4, University of California Press, Berkeley, pp. 223–239.

Eigen, M. & Schuster, P. (1979), *The Hypercycle: A Principle of Natural Self-Organization.*, Springer.

Endy, D. (2005), 'Foundations for engineering biology', *Nature* **438**, 449–453.

Fermi, E., Pasta, J. & Ulam, S. (1955), 'Studies of nonlinear problems', *LASL Report LA-1940* **5**. reprinted in the collected work of E. Fermi (University of Chicago, Chicago, 1965), vol. 2, pp. 977–988, 1965.

Fisher, M., Malcolm, G. & Paton, R. (2000), 'Spatio-logical processes in intracellular signalling', *BioSystems* **55**, 83–92.

Fontana, W. (1992), Algorithmic chemistry, *in* C. G. Langton, C. Taylor, J. D. Farmer & S. Rasmussen, eds, 'Proceedings of the Workshop on Artificial Life (ALIFE '90)', Vol. 5 of *Santa Fe Institute Studies in the Sciences of Complexity*, Addison-Wesley, Redwood City, CA, USA, pp. 159–210.

Fontana, W. & Buss, L. W. (1994), "'the arrival of the fittest": Toward a theory of biological organization', *Bulletin of Mathematical Biology* .

Giavitto, J.-L. (2003), Topological collections, transformations and their application to the modeling and the simulation of dynamical systems, *in* 'Rewriting Technics and Applications (RTA'03)', Vol. LNCS 2706 of *LNCS*, Springer, Valencia, pp. 208 – 233.

Giavitto, J.-L., Godin, C., Michel, O. & Prusinkiewicz, P. (2002), *Modelling and Simulation of biological processes in the context of genomics*, Hermes, chapter "Computational Models for Integrative and Developmental Biology".

Giavitto, J.-L., Malcolm, G. & Michel, O. (2004), 'Rewriting systems and the modelling of biological systems', *Comparative and Functional Genomics* **5**, 95–99.

Giavitto, J.-L. & Michel, O. (2001), Declarative definition of group indexed data structures and approximation of their domains, *in* 'PPDP '01: Proceedings of the 3rd ACM SIGPLAN international conference on Principles and practice of declarative programming', ACM Press, New York, NY, USA, pp. 150–161.

Giavitto, J.-L. & Michel, O. (2002*a*), Data structure as topological spaces, *in* 'Proceedings of the 3nd International Conference on Unconventional Models of Computation UMC02', Vol. 2509, Himeji, Japan, pp. 137–150.

Giavitto, J.-L. & Michel, O. (2002*b*), 'Pattern-matching and rewriting rules for group indexed data structures', *ACM SIGPLAN Notices* **37**(12), 76–87.

Giavitto, J.-L. & Michel, O. (2002*c*), 'The topological structures of membrane computing', *Fundamenta Informaticae* **49**, 107–129.

Giavitto, J.-L. & Michel, O. (2003), 'Modeling the topological organization of cellular processes', *BioSystems* **70**(2), 149–163.

Giavitto, J.-L., Michel, O. & Cohen, J. (2002), Pattern-matching and rewriting rules for group indexed data structures, *in* 'ACM Sigplan Workshop RULE'02', ACM, Pittsburgh, pp. 55–66.

Giavitto, J.-L. & Spicher, A. (2008*a*), *Systems Self-Assembly: multidisciplinary snapshots*, Elsevier, chapter Simulation of self-assembly processes using abstract reduction systems, pp. 199–223. doi:10.1016/S1571-0831(07)00009-3.

Giavitto, J.-L. & Spicher, A. (2008*b*), 'Topological rewriting and the geometrization of programming', *Physica D* **237**, 1302–1314.

Gibson, M. C., Patel, A. B., Nagpal, R. & Perrimon, N. (2006), 'The emergence of geometric order in proliferating metazoan epithelia', *Nature* **442**, 1038–1041.

Gillespie, D. T. (1977), 'Exact stochastic simulation of coupled chemical reactions', *J. Phys. Chem.* **81**(25), 2340–2361.

Greenberg, J. & Hastings, S. (1978), 'Spatial patterns for discrete models of diffusion in excitable media', *SIAM Journal on Applied Mathematics* pp. 515–523.

Itkis, Y. (1976), *Control Systems of Variable Structure*, Wiley.

Jansson, P. & Jeuring, J. (1997), PolyP - a polytypic programming language extension, *in* 'Principles of Programming Languages', ACM Press, pp. 470–482.

Knight, T. (2006), 'Idempotent vector design for standard assembly of biobricks', *MIT Synthetic Biology Working Group* .

Lindenmayer, A. (1968*a*), 'Mathematical models for cellular interaction in development, Parts I and II.', *Journal of Theoretical Biology* **18**, 280–315.

Lindenmayer, A. (1968*b*), 'Mathematical models for cellular interaction in development, Parts I and II.', *Journal of Theoretical Biology* **18**, 280–315.

Luisi, P. L. (2003), 'Autopoiesis: a review and a reappraisal', *Naturwissenschaftenl* (90), 49–59.

Lynch, J. (2008), A Logical Characterization of Individual-Based Models, *in* 'Logic in Computer Science, 2008. LICS'08. 23rd Annual IEEE Symposium on', pp. 379–390.

McAdams, H. & Shapiro, L. (1995), 'Circuit simulation of genetic networks', *Science* **269**(5224), 650.

Michel, O., Spicher, A. & Giavitto, J.-L. (2008), 'Rule-based programming for integrative biological modeling – application to the modeling of the λ phage genetic switch', *Natural Computing* .

Munkres, J. (1984), *Elements of Algebraic Topology*, Addison-Wesley.

Păun, G. (2001), 'From cells to computers: computing with membranes (P systems)', *Biosystems* **59**(3), 139–158.

Shnerb, N. M., Louzoun, Y., Bettelheim, E. & Solomon, S. (2000), 'The importance of being discrete: Life always wins on the surface', *PNAS* **97**(19), 10322–10324.

Smith, J. (1999), *Shaping life: genes, embryos and evolution*, Yale University Press.

Spicher, A., Fats, N. & Simonin, O. (2009), From reactive multi-agents models to cellular automata, *in* 'International Conference on Agents and Artificial Intelligence'.

Spicher, A. & Michel, O. (2005), Using rewriting techniques in the simulation of dynamical systems: Application to the modeling of sperm crawling, *in* 'Fifth International Conference on Computational Science (ICCS'05), part I', Vol. 3514 of *LNCS*, Springer, Atlanta, GA, USA, pp. 820–827.

Spicher, A. & Michel, O. (2007), 'Declarative modeling of a neurulation-like process', *BioSystems* **87**(2-3), 281–288.

Turing, A. M. (1952), 'The chemical basis of morphogenesis', *Phil. Trans. Roy. Soc. of London* **Series B: Biological Sciences**(237), 37–72.

Turk, G. (1991), Generating textures for arbitrary surfaces using reaction-diffusion, *in* T. W. Sederberg, ed., 'Computer Graphics (SIGGRAPH '91 Proceedings)', Vol. 25, pp. 289–298.

Varela, F. J., Maturana, H. R. & Uribe, R. (1974), 'Autopoiesis: the organization of living systems, its characterization and a model', *Autopoiesis: the organization of living systems, its characterization and a model* **5**, 187–196.

von Dassow, G., Meir, E., Munro, E. & Odell, G. (2000), 'The segment polarity network is a robust developmental module', *Nature* **406**(6792), 188–192.

Von Neumann, J. (1966), *Theory of Self-Reproducing Automata*, Univ. of Illinois Press.

Wolfram, S. (1986), *Theory and applications of cellular automata*, Advanced Series on Complex Systems, Singapore: World Scientific Publication.

Woolridge, M. & Wooldridge, M. (2001), *Introduction to multiagent systems*, John Wiley & Sons, Inc. New York, NY, USA.

Zhang, X., De Cock, K., Bugallo, M. F. & Djuric, P. M. (2003), Stochastic simulation and parameter estimation of enzyme reaction models, *in* 'IEEE Workshop on Statistical Signal Processing'.

An MGS Implementation of the Runge-Kutta Methods

The Runge-Kutta methods is a famous and widely used family of integration scheme for solving problems of the form:

$$\frac{d\mathbf{X}(t)}{dt} = f(\mathbf{X}(t),t) \qquad \mathbf{X}(0) = X_0$$

They are based on the techniques developed by C. Runge and M.W. Kutta at the beginning of the XX$^{\text{th}}$ century.

In the following, we propose an MGS implementation of the classical explicit fourth order Runge-Kutta method (RK4). This example can be extended to most of other Runge-Kutta methods. As the Euler method, the RK4 allows the computation of a sequence of vectors $\mathbf{X}_n$ with a more complex scheme reducing errors of approximations. Starting from an initial state $\mathbf{X}_0$ at time t_0, a step a the RK4 is given by

$$\mathbf{X}_{n+1} = \mathbf{X}_n + \frac{\Delta t}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4) \qquad t_{n+1} = t_n + \Delta t$$

with

$$\mathbf{k}_1 = f(\mathbf{X}_n, t_n) \qquad \mathbf{k}_2 = f(\mathbf{X}_n + \frac{\Delta t \mathbf{k}_1}{2}, t_n + \frac{\Delta t}{2})$$
$$\mathbf{k}_3 = f(\mathbf{X}_n + \frac{\Delta t \mathbf{k}_2}{2}, t_n + \frac{\Delta t}{2}) \qquad \mathbf{k}_4 = f(\mathbf{X}_n + \Delta t \mathbf{k}_3, t_n + \Delta t)$$

The MGS implementation can be divided into 3 steps:

1. the computation of expressions $\mathbf{X}_n + c\mathbf{k}$ where c is a coefficient (either $\frac{\Delta t}{2}$ or Δt) and $\mathbf{k}$ takes the value $\mathbf{k}_i$,
2. the computation of $\mathbf{X}_{n+1}$ knowing the $\mathbf{k}_i$, and
3. the composition of the two first steps.

Steps (1) and (2) are straightforward and can be implemented as follows:

```
trans RK4a[c,k] = { x => x + c*k.(^x) }
trans RK4b[dt,k1,k2,k3,k4] = {
  x => x + dt/6( k1.(^x) + 2*k2.(^x) + 2*k3.(^x) + k4.(^x) )
}
```

These transformations apply the required computations on each coordinate of `X`. The final step consists in implementing the function `RK4`:

```
fun RK4[dt,f](X,t) = (
  let k1 = f(X,t) in
  let k2 = f(RK4a[c=dt/2,k=k1](X),t+dt/2) in
```

```
  let k3 = f(RK4a[c=dt/2,k=k2](X),t+dt/2) in
  let k4 = f(RK4a[c=dt,k=k3](X),t+dt) in
    (RK4b[dt=dt,k1=k1,k2=k2,k3=k3,k4=k4](X),t+dt)
)
```