Integrated Regulatory Networks (IRNs): Spatially organized biochemical modules

Jean-Louis Giavitto, Hanna Klaudel, Franck Pommereau

Integrated Regulatory Networks (IRNs): spatially organized biochemical modules

Jean-Louis Giavitto[b], Hanna Klaudel[a], Franck Pommereau[a]

[a] *IBISC, University of Évry and Genopole*
Tour Évry 2, 523 place des terrasses de l'Agora, 91000 Évry, France
[b] *IRCAM - UMR STMS 9912 CNRS*
4 place Igor Stravinsky, 75004 Paris, France

Abstract

In this paper, we aim at modeling and analyzing the regulation processes in multi-cellular biological systems, in particular tissues. The modeling framework is a generalization of several existing formalisms. In particular, it can be seen as an extension of logical regulatory networks (*à la* Thomas) with information about cells physical state and environment, *e.g.*, their spatial relationships. The resulting formalisms, called *integrated regulatory networks* (*IRNs*) is equipped with a transition systems semantics that preserves the possibility of an enumerative and exhaustive state space exploration. This paper presents the modeling framework, its semantics, as well as a prototype implementation that allowed preliminary experiments on some applications related to biology.

Keywords: Logical regulatory networks, topological collections, enumerable state space, spatial modeling, systems biology.

1. Introduction

Regulation processes are the corner stone to understand many aspects of biological systems. They occur at many levels like transcription and translation of the genetic material, or protein modifications. They define complex networks of interactions that cannot be easily understood without resorting

Email addresses: jean-louis.giavitto@ircam.fr (Jean-Louis Giavitto), klaudel@ibisc.univ-evry.fr (Hanna Klaudel), pommereau@ibisc.univ-evry.fr (Franck Pommereau)

to formal modeling and automated analysis. The *generalized logical formalism* initially proposed by René Thomas in the seventies [41, 42, 43], is a discrete modeling formalism that has proved to be an effective way to capture regulation processes and analyze them. It has been successfully applied to the study of a variety of regulatory networks comprising relatively large number of components [35, 36]. This formalism however does not provide any modeling device to specifically address the question of multi-cellular systems, where the regulatory networks of cells can influence each other in a way that is dependent on the spatial relationships between the cells.

On the other hands, several formalisms inspired by formal language theory have been used to model and simulate the dynamics of multicellular systems. For instance, *Lindenmayer systems* are generative grammars that have been successfully used since the seventies in the modeling of plant growth and developmental biology [29, 34]. This framework is widespread for the realistic simulation of developing organisms, but has not been the target of automated analysis like model-checking.

This paper aims at proposing a modeling framework integrating regulation *and* development in multi-cellular systems. It takes into account physical information about cells (like weight, volume or spatial organization) as well as about the environment (like temperature or mechanical stress). We are particularly interested by the modeling of the dynamics of tissues and, following the central dogma of molecular biology [12], the physical transformations of cells (like migration, division and apoptosis – cell death) will be governed by the regulatory processes.

This framework is devoted to the analysis of systems such as developmental processes, invasive cancers, plant growth, etc. Therefore, one of our main objectives in such a modeling framework is to preserve the ability to perform model-checking based analysis in order to be able to assess causality-related properties and reveal rare events, which usually cannot be obtained through simulation. This constrains the possible choices for modeling physical information. In particular, there should be a finite number of possible evolutions from a given configuration and they should be enumerable. Moreover, each configuration should be represented in a normalized form, allowing the recognition of two identical configurations. For instance, floating-point positions are not a possible solution; instead, we shall use solutions based on discrete (qualitative) and combinatorial structures.

Contributions. Our starting point is a qualitative modeling approach developed in [9, 10] based on logical regulatory networks [41, 42, 43] and extended later to introduce modularity [7, 8]. In this extension, modularity allows a systematic modeling of multi-cellular systems, each cell being represented by a *module*. However, the spatial relationships between the modules within a model is represented in a very coarse way, with no link to any kind of geometrical or topological information. Moreover, in this setting, the number of modules is fixed when the model is first built. In [15], we extended this by including explicitly the spatial aspects of a system and allowing spatial transformations through explicit cells division (by duplication), apoptosis (by destruction) or migration. Finally, only two kinds of discrete space representation were considered and the interactions between the logical (regulatory) specification and the spatial specification of the system were rather *ad-hoc*.

In the present paper, we further generalize and unify these approaches by considering physical aspects and not only spatial ones. Physical aspects include arbitrary transformations as well as environmental factors which are not necessarily controlled by the regulatory processes (*e.g.*, the temperature) but affect the model dynamics (*e.g.*, the rate of diffusion).

Spatial aspects are the physical aspects concerned with the topological or geometrical organization of the system. Their modeling is specified through rewriting rules, called transformations, acting on *topological collections* [14]. Topological collections generalize the notion of labeled graphs to higher dimensional objects. This framework, implemented in the language MGS, has been used in the context of P systems [17] and in several large modeling projects in systems biology [2, 18, 39].

The resulting framework, called *IRNs* for *Integrated Regulatory Networks*, allows the specification and the analysis of spatially organized biochemical modules. It is given a semantics in terms of transition systems that is implemented in a prototype, allowing to prove the feasibility of the approach and to run preliminary experiments on simple applications. We present in particular two instances of spatial information specification. One is based on sequences that can be extended or shrunk and is used to specify and analyze a model for the cell division and differentiation of a filamentous blue-green alga *Anabæna catenula* [46, 23]. The other one is based on predefined grids and applied to the modeling of a quorum sensing mechanism.

Notice that the approach presented in this paper is applicable only if a unique sort of module is considered at the same time, *i.e.*, if all the cells in a tissue are of the same kind. The extension to take into account multi-sorted

systems is quite straightforward, but the resulting notations are much more complex. Our prototype implementation actually does not have any such limitation. However, for this paper, an intuitive and simpler presentation has been preferred. Furthermore, this paper does not consider the expression and analysis of properties in the models, we simply provide a framework with enumerable state spaces which is enough to guarantee that any model-checking technique is potentially applicable (in particular, modal/temporal logic based approaches).

Outlines. The next section introduces the background of our work, in particular the modeling formalisms we start from (for regulatory aspects as well as for the modeling of spatial information). Our contributions are then presented in sections 3 and 4, the former is dedicated to the definition of the syntax and semantics of IRNs, while the latter presents some experiments. The paper ends with a conclusion and a discussion about future works.

2. Background

In this section we present some background material about logical regulatory networks and about topological collections and their transformations. These formalisms will be used in the definition of the IRN framework and will be illustrated through an example: the modeling of the growth of a blue-green alga.

2.1. A running example

Anabæna catenula is a cyanobacterium that grows in filaments of one hundred cells or more. The sequence of cells is composed of small and large cells that are polarized to the right or to the left.

In case of nitrogen starvation, a differentiation process occurs: specialized cells called *heterocysts* differentiate from the photosynthetic vegetative cells. Heterocysts are anaerobic factories for nitrogen fixation and are located at regular intervals along each filament.

Plant signals exert both positive and negative regulatory control on heterocyst differentiation. Wilcox *et al.* have proposed an activator-inhibitor model of heterocysts differentiation where the high concentration of the activator triggers the heterocysts differentiation [46]. The production of the activator is an autocatalytic reaction and also catalyzes the production of the inhibitor. The inhibitor is an antagonist substance that represses the

activity of the activator when its concentration is high enough. The diffusion of the inhibitor to the neighboring cells prevents neighbors becoming heterocysts and explains why heterocysts appear in a regular spaced pattern in the filament. A possible IRN model of the growth and differentiation processes of *Anabæna* will be presented in section 3.2.

2.2. Logical regulatory networks

A logical *regulatory network* [41, 42, 43] is usually depicted as a graph whose vertexes are *regulatory components*, for example, genes or proteins, and whose arcs indicate how each component is influenced by others. A simple regulatory network for an *Anabæna* cell is depicted in the middle of Figure 1. Component Sz encodes the size of the cell (0 for small, 1 for big); component Hc encodes whether the cell is heterocyst (with value 2), vegetative (value 0) or in an intermediary state (value 1). The intuition of the network depicted this way is that Hc is auto-regulated and inhibits Sz. This is thus a network of influences, not to be confused with an automaton nor with the corresponding state space. Different shapes of arrows heads indicate activation ($\rightarrow$) or inhibition ($\dashv$).

More formally, a regulatory network is defined as a set of regulatory components, each component Cmp being associated with a *regulatory function* $next_{\mathsf{Cmp}}$ that provides the following information: its codomain defines the range of values Cmp can assume, the current value of Cmp being denoted as x_{Cmp}; its arguments define the regulatory components Cmp depends on; its evaluation defines the level toward which Cmp is called to evolve by steps of ± 1. For example, in Figure 1, Hc is always called to evolve towards 2, but starting from 0, it must go through 1.

$$dom(\mathsf{Hc}) \stackrel{\text{df}}{=} \{0,1,2\}$$

$$next_{\mathsf{Hc}}(x_{\mathsf{Hc}}) \stackrel{\text{df}}{=} 2$$

$$dom(\mathsf{Sz}) \stackrel{\text{df}}{=} \{0,1\}$$

$$next_{\mathsf{Sz}}(x_{\mathsf{Sz}}, x_{\mathsf{Hc}}) \stackrel{\text{df}}{=} \begin{cases} x_{\mathsf{Sz}} & \text{if } x_{\mathsf{Hc}} > 0 \\ 1 & \text{if } x_{\mathsf{Hc}} = 0 \end{cases}$$

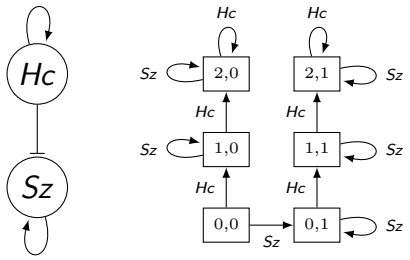

Figure 1: Left: the formal definition of a simple regulatory network for one cell of *Anabæna*. Middle: its graphical abstract representation. Right: the corresponding state space where each node is a state labeled by $x_{\mathsf{Hc}}, x_{\mathsf{Sz}}$ and each edge corresponds to the change of one variable as indicated in its label.

A *state* s of a regulatory network is a *binding* that provides for each component Cmp its current level $s(\mathsf{Cmp}) \stackrel{\text{df}}{=} x_{\mathsf{Cmp}}$. In other words, a binding is a partial function on a finite domain associating values to components. Given a state s and a component Cmp, it is possible to evaluate $next_{\mathsf{Cmp}}$, yielding a target value ${x_{\mathsf{Cmp}}}'$ for Cmp. If ${x_{\mathsf{Cmp}}}' \neq x_{\mathsf{Cmp}}$, this defines a possible *evolution* of the system to a state s' that is such that $s'(\mathsf{C}) \stackrel{\text{df}}{=} s(\mathsf{C})$ for each component $\mathsf{C} \neq \mathsf{Cmp}$, and:

$$s'(\mathsf{Cmp}) \stackrel{\text{df}}{=} \begin{cases} x_{\mathsf{Cmp}} + 1 & \text{if } {x_{\mathsf{Cmp}}}' > x_{\mathsf{Cmp}} \ , \\ x_{\mathsf{Cmp}} - 1 & \text{if } {x_{\mathsf{Cmp}}}' < x_{\mathsf{Cmp}} \ . \end{cases}$$

Such an evolution rule allows to define the *state space* that is suitable to perform model-checking of various reachability- or causality-related properties. The right part of Figure 1 shows the state space of the regulatory model of *Anabæna*.

2.3. Topological collections and topological rewriting

Integrative modeling of biological processes (*e.g.*, in system biology) relates different models that operate on different levels of abstraction and various spatial and time scales. The spatial organization of cells is crucial and the description of the morphogenetic processes at a cellular level implies the integration of molecular mechanisms such as cell-cell signaling, mechanical stresses and genetic regulation embedded in a complex dynamic geometry [11].

Topological collections have been introduced in [17] to describe arbitrary complex spatial structures that appear in biological systems [18] and other dynamical systems with a time varying structure [14, 22]. Figure 2, adapted from [40], summarizes the main spatial representations actually used in biological modeling (see also [5] for a review). The notion of topological collection has been used successfully to represent all these kinds of space.

A topological collection is a weakening of the notion of *topological chain* that is developed in algebraic topology and corresponds to a labeled *cellular complex* [31]. An (abstract) cellular complex is a formal construction that builds a space in a combinatorial way through more simple objects called *topological cells*.[1] Each topological cell abstractly represents a part of the

[1]The reader must pay attention not to confuse biological and topological cells.

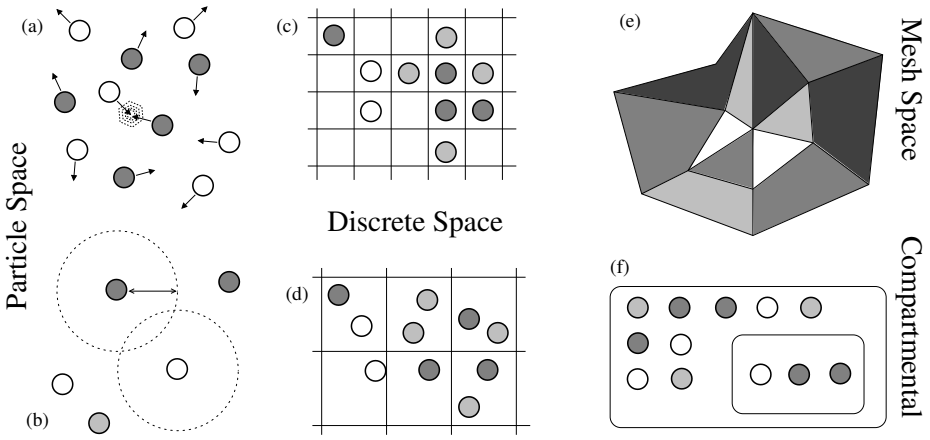

Figure 2: Various representation of space often used in systems biology according to [40]. (a) In particle space, molecules are represented as individual particles with positions in a continuous space. (b) Such methods accommodate a discrete dynamics when particles jump in time and space by calculating the maximum distance that the particle can travel and interact in the time slot. Lattices discretize regularly the space: in microscopic lattices (c) at most one particle is allowed to occupy a lattice site while mesoscopic methods (d) allow several ones. (e) Mesh spaces are usually used to solve PDE (*e.g.*, in reaction-diffusion systems) or for cellular automata like models. (f) Compartmental spaces focus on the molecular transfers between compartments.

whole space. The structure of the whole space, corresponding to the partition into topological cells, is considered through the *incidence relationships*, relating a cell and the cells in its boundary. A topological chain is a function from a cellular complex to a set of labels equipped with some algebraic structure [31].

Transformations of topological collections are defined by rewriting rules. A general notion of topological collection rewriting [14] has been developed in a logical style [17] and in a more operational style [38].

2.3.1. Topological collections

In this paper, we focus on a particular case of topological collections, namely on graphs. We can then forget some of the full technical machinery for topological collections manipulation, the target topological collection downgrading to a labeled graph. Let $\mathbb{V}$ be a set of symbols used to denote the vertexes and $\mathbb{E}$ a set of symbols used to denote the edges of the graph. For technical reasons, an edge is not an element of $\mathbb{V} \times \mathbb{V}$: the connection between vertexes and edges are captured by a strict partial order (*i.e.*, an

irreflexive, transitive and antisymmetric binary relation) on the set of topological cells $\mathbb{C} \stackrel{\text{df}}{=} \mathbb{V} \cup \mathbb{E}$, written $\prec_{\mathbb{C}}$. We have $v \prec_{\mathbb{C}} e$ if vertex v is one of the ends of edge e. A one dimensional *abstract cellular complex* $K = (A, \prec)$ on $\mathbb{C}$ is a partially ordered subset of $(\mathbb{C}, \prec_{\mathbb{C}})$ and $\prec$ is called the *incidence relationships* of the complex K.

A *topological collection* over $K \stackrel{\text{df}}{=} (A, \prec)$ with labels in $\mathbb{L}$ (the set of possible labels) is a triple $(K, C, \mathbb{L})$ where C is a partial function from $A \subset \mathbb{C}$ to $\mathbb{L}$. For convenience, such a collection is denoted by C. Moreover, $|C|$ denotes the set of cells $c \in A$ for which $C(c)$ is defined, and $vert(C) = A \cap \mathbb{V}$ is the set of the vertexes of C.

The topological collection C can be written as a formal sum $\sum_{c \in |C|} \ell_c \cdot c$ where $\ell_c \stackrel{\text{df}}{=} C(c)$. With this notation, K and $\mathbb{L}$ are left implicit but can usually be recovered from the context. By convention, when we write a collection C as a sum $C \stackrel{\text{df}}{=} \ell_1 \cdot c_1 + \cdots + \ell_p \cdot c_p$, we insist that all c_i are distinct. Notice that the addition is associative and commutative.

2.3.2. Topological rewriting

In the framework of this paper, topological rewriting can be defined as a simple kind of graph rewriting following an approach similar to that taken in [32]: using the additive representation of topological collections, topological rewriting can be simply defined as an adapted version of conditional first-order associative-commutative term rewriting, see [38] for the details. The formal definition of topological rewriting is less interesting than the syntax of the pattern language used to specify the left hand side (lhs) of a rewriting rule: as a matter of fact, the lhs of a rule must match a sub-collection, that is a subset of $(K, C, \mathbb{L})$. This includes a sub-relation of the incidence relation $\prec$ of K. This information can be difficult to specify without the help of a dedicated language.

We rely here on the syntax of the *path pattern language* defined for MGS [20]. A transformation T is a function specified by a set of rewriting rules $\{p_1 \to e_1, \ldots, p_n \to e_n\}$ where each p_i is a pattern and each e_i is an expression. An application of such a rule matches a sub-collection with one p_k that is then substituted by the result of expression e_k. The path pattern language relies on the constructions below.

- A *label* $\ell \in \mathbb{L}$ matches a vertex labeled by this value. So, it corresponds to the term $\ell \cdot {}^\wedge v$ where ${}^\wedge v$ is a fresh variable ranging over the vertexes.

- A *pattern variable* v matches a vertex and its label. The identifier v can be used elsewhere in the rule to refer to the label of the matched cell; the cell itself can be referred through the special identifier $\hat{}v$. Using the additive notation for topological collections, this pattern is translated to $v \cdot \hat{}v$ where v ranges over the labels, and variable $\hat{}v$ ranges over the vertexes.

- A *guard* / can be used to specify a condition that must be satisfied by the matching. For instance, expression $v/v > 5$ matches a cell $\hat{}v$ labeled by a integer greater than 5.

- Associative operator "," is used to specify a path, *i.e.*, a sequence of elements. A comma implies also some constraints on the incidence relationships linking the two arguments: in the additive notation, the pattern v, w translates to the conditional pattern

$$v \cdot \hat{}v + w \cdot \hat{}w \;/\; \exists \hat{}u : \; \hat{}v \prec \hat{}u \;\wedge\; \hat{}w \prec \hat{}u$$

  In other words, the vertexes matched by v and w are linked by edge u.

Rule applications are controlled through a *rule application strategy*. Several strategies are available in MGS like the maximal parallel application used in Lindenmayer or P systems, and the Gillespie stochastic simulation algorithm used in the exact simulation of chemical reactions [37]. These strategies are non-deterministic, *i.e.*, applied on a collection C, only one of the possible outcomes (randomly chosen) is returned by the transformation. We will see later on how our implementation overcomes this limitation in order to explore the whole state space.

2.3.3. Modelling the growth of Anabæna

In this section, we apply topological collections to the modeling of the growth of *Anabæna*. In MGS, several specialized types of topological collections are available for the modeler as predefined data types. Here we will use *sequences* to model the one dimensional organization of the alga. Sequences are a particular kinds of topological collections where the underlying graph are linear, *i.e.*, the set of vertexes $\mathbb{V}$ is equipped with a total order $<$. In addition, this total order is compatible with $\prec$, that is, if there exists an edge e such that $v \prec e$ and $v' \prec e$, then $v < v'$ or $v' < v$. Edges are directed and the direction is taken into account by the comma operator.

We use the four symbols: left, right, Left and Right, to represent the various states of polarity and size of an *Anabæna* cell. The following four transformation rules give the fate of each kind of cell [30]:

$$\mathsf{left} \to \mathsf{Left} \qquad \mathsf{Left} \to \mathsf{left}, \mathsf{Right} \qquad \mathsf{right} \to \mathsf{Right} \qquad \mathsf{Right} \to \mathsf{Left}, \mathsf{right}$$

meaning that a small left polarized cell grows into a big left polarized cell, a big left polarized cell divides into a small cell with the same orientation, and a big cell with opposite orientation, etc. Note that the comma operator is overloaded and appears also in the right hand side (rhs). In this context, it is simply used to build the new sequence to substitute. Thus, irrespectively of the complexity of the sub-collection to be substituted and the complexity of the underlying spatial organization, the lhs and rhs of a rule handle simple paths (sequences).

The "one symbol = one cell state" approach for the modeling of *Anabæna* is no longer acceptable when the number of possible states increases. So, we shall use arbitrary values to label the topological cells of a collection and we illustrate here the use of MGS *records* to handle concisely a state as a product of sub-states. A record is a dictionary associating values to names, *i.e.*, it implements a binding (the fields of the record form the domain of the binding). For example, we can use a record with two fields to specify the polarity and the size of a cell:

$$\{\ \text{Polarity} = \mathsf{left};\ \text{Sz} = \mathsf{small}\ \}$$

Curly brackets are used in MGS to delimit the record r; application $r(a)$ to access the value of a field. Special constructions can be used to ease the pattern-matching of such objects. In particular, a pattern

$$\{\ \text{Polarity} = \mathsf{left};\ \text{Sz} = \mathit{size}\ \}\ \text{as}\ r$$

matches any record with at least a field "Sz" and a field "Polarity" with value left. The construction "as" is used to give a name to the matched record. This identifier, and the identifier used to match the value of a field, can be used elsewhere in the rule. For example, the two previous rules used to increase the size of left and right polarized cell, can be summarized by a single rule:

$$\{\ \text{Sz} = \mathsf{small}\ \}\ \text{as}\ r \to r + \{\ \text{Sz} = \mathsf{big}\ \}$$

The + operator in the rhs denotes the asymmetric merge of records [33]. The expression $r_1 + r_2$ computes a new record r having the fields of both r_1 and

r_2: $r(a)$ has the value of $r_2(a)$ if the field a is present in r_2, otherwise it has the value of $r_1(a)$. We take the same notations for bindings.

It is possible to recover various kinds of rewriting using dedicated incidence relationships: set and multiset rewriting (*e.g.*, as in P systems), string rewriting (*e.g.*, as in Lindenmayer systems) or array rewriting (*e.g.*, as in lattice gaz automata). However, if topological rewriting is very expressive for simulation purposes, the automated analysis of the behavior of the iterated applications of topological rewriting rules is very difficult. These difficulties are partly caused by the wide range of rewriting strategies that can be used and by the intertwining between the management of the labels (logical part) and of the structure (physical part).

So, the obvious follow-up was to combine the logical and the physical parts of the model and to establish a uniform presentation of both aspects, especially from the point of view of the control, in order to ease the analysis of such models. This is proposed in the IRN formalism.

3. Integrated regulatory networks

3.1. Intuitive presentation

Integrated regulatory networks (*IRNs*), an extension of logical regulatory networks, are presented in several steps. First, logical regulatory networks are generalized by removing the condition that component value domains are integer intervals $[0; n]$ and that they evolve by steps of ± 1. Then, we introduce the notion of *modules* as in [8, 15]. This results in the definition of *local variables* and their update functions. For example, local variables may be regulatory components or modules characteristics like weight or size. To model regulation, it is of course possible to resort to local variables whose domains are integer intervals $[0; n]$ and that actually evolve by ± 1.

The evolution of a local variable within a module may depend on the values of some local variables in other modules. This is modeled as a *local measure*, allowing to collect those values in the neighborhood of the evolving module. For example, a local measure may integrate the concentrations of a regulatory component diffused toward a cell from its neighbors according to the distances between cells.

Next, to model spatial relations between modules, we *localize* them on a topological collection. Neighborhood relationships between modules are represented by a labeled graph. For example, the graph may reflect the spatial arrangement of cells in a tissue, or the positions of bacterias in a

population. The vertexes of this graph are module identifiers and a vertex and its associated label correspond to a given module. Two modules i and j are neighbors if there is an edge between the vertexes labeled i and j. Topological collections are used to implement a database that records the neighborhood relationships, which can be queried and updated efficiently. So, local variables become bindings attached to the vertexes of the collection. Similarly, local measures taken from a module i become a computation of a single value from (possibly several) multisets of pairs $(\ell_j, x_{\mathsf{lvar}\,j})$ where ℓ_j is the label of the arc between i and j and $x_{\mathsf{lvar}\,j}$ is the value of a local variable *lvar* in module j. Local measures are actually generalizations of *integration functions* from [8].

The whole graph may be labeled itself, yielding *global variables*. Global variables may reflect environmental parameters like temperature or pressure. Moreover, *global measures* are obtained by computing a value from an observation of the whole graph. For example, a global measure may be the number of cells in the tissue, or the average of a local variable over the cells.

Finally, we introduce *graph updates* to allow for evolutions of the structure of the graph. They correspond to physical modifications in the system, for instance a reconfiguration of the spatial relationship like in cells migration, but also in cells division or death.

3.2. Anabæna *example*

In this section, we will sketch the use of the IRN formalism to the modeling of the growth of *Anabæna* and the cell differentiation process described in section 2.1. This example implies a fundamental mechanism in development: a morphogenesis driven by a reaction-diffusion process taking place in a growing media.

The first model allowing an extensive simulation of this process was developed in the field of parametric L systems [24]. It is based on the numerical resolution of a set of coupled differential equations that specify the diffusion and the reaction of the two morphogens amongst the cells (considered as homogeneous compartments). The cell division process (which introduce additional equations to solve at each division) is specified using an L system grammar where each symbol is labeled by a set of continuous variables (concentrations, size, etc.).

In contrast to this model, we propose here a discrete and qualitative model of *Anabæna* growth and differentiation. Our goal is not to compare this model with parametric L systems, but to investigate the usability of

model-checking techniques in the analysis of developmental processes. For example, one of our goals is to check that our model cannot produce two neighbor heterocysts.

The system and its evolutions are formalized as follows. First, we assume that *Anabæna* cells are linked in one sequence containing a number *Nbr* of cells (a global measure), the maximal number of cells being limited to Nbr_m in order to ensure the finiteness of the dynamics (the corresponding transition system).

The environment can provide nitrogen (modelled by a global variable *Nitro*) to sustain the growth of the population. This variable is simply controlled by the number of cells in the system, *i.e.*, if *Nbr* is below a fixed threshold Nbr_N then *Nitro* remains constant, otherwise it decreases.

A cell is characterized by two local variables: its size *Sz* and its kind *Hc* (for the sake of simplicity, we forget the polarity). The size *Sz* can be small or big. The kind *Hc* belongs to {vegetative, undetermined, heterocyst}, where undetermined is an intermediary state in the transition between the vegetative and heterocyst states. If there is some nitrogen, cells cannot become heterocysts. There is nitrogen if it is provided by the environment (*i.e.*, *Nitro* is non zero) or because there are heterocysts in the neighborhood: this property is modeled as a local measure *Ah*, a Boolean function indicating whether a neighbor is heterocyst (*Ah* stands for "Any heterocyst?"). A small cell can grow if there is some nitrogen or if there are heterocysts in the neighborhood. A big cell divides under the same conditions (and if the population is less than Nbr_m) to give two small children cells. The latter is modeled as a graph update *Div*. If there is no more nitrogen and if there is no heterocyst in the neighborhood, a cell can change its kind from vegetative to undetermined, and then to heterocyst. A heterocyst remains heterocyst as long as there is a starvation of nitrogen. A heterocyst does not grow and cannot divide. Notice that since we do not model cells death in this example (for simplicity), the population will never decrease. So, whenever a nitrogen starvation occurs, it becomes permanent.

The expected dynamics is formally defined by the evolution function specified in the tables given in Figure 3. We will see later on how division *Div* is specified.

3.3. Formal specification of the framework

An *integrated regulatory network* (*IRN*) is specified by the finite set of its global and local variables with their variable update functions, its local

current *Nitro*	*Nbr*	
	$< Nbr_N$	$\geq Nbr_N$
0	0	
any $n > 0$	n	$n-1$

current *Hc*	*Nitro*		
	0		> 0
	Ah		
	false	true	
vegetative	undetermined	vegetative	*vegetative*
undetermined	heterocyst	undetermined	
heterocyst	heterocyst		

current *Sz*	*Hc*			
	heterocyst	vegetative or undetermined		
		Nitro		
		0		> 0
		Ah		
		false	true	
small	small	small	big	big
big	big			

Figure 3: Evolution tables of variables: *Nitro* (top), *Hc* (middle) and *Sz* (bottom).

and global measures definitions, and its graph update functions. A *state of an IRN* is defined by a binding λ of its global variables and a topological collection C that will define, in particular, the bindings for the vertexes (local variables) and the edges.

3.3.1. Syntactical aspects

To support intuition, IRNs are depicted using the conventions described in Figure 4. The shape and style (dotted or plain line) of the components denote their nature while the arcs indicate a potential influence (resulting in corresponding arguments in the function). Note that only one kind of arrow tip is used because influences may be arbitrary and not limited to inhibition or activation, as in logical regulatory networks. The constraints about the arcs are defined consistently with the arguments allowed for each kind of update function. The rationale is as follows: local variables and measures can depend both on local and global information that is available to every module; global variables or measures occur globally in the system and so

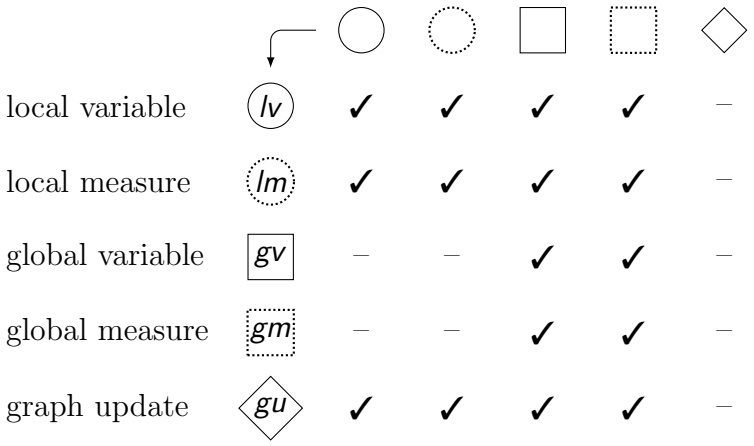

Figure 4: Graphical conventions: measures are depicted in dotted lines; local objects are depicted by round nodes while global objects are depicted by square nodes. The check marks indicate whether an arc is allowed (✓) or not (–) toward the node types in the left column from each node type in the top row.

cannot depend on any local information; graph updates occur at some given module and thus can depend on local or global information, but no function can depend on a graph update because it does not compute any value but instead transforms the spatial structure (*i.e.*, C). Moreover, we assume that any variable may depend on its current value, so we do not need to draw self-loops in the graphical representation. Finally, we require that an IRN is *well-formed*, in the sense that there is no mutual recursion between measures and all functions are total on their finite domain and computable. Using these conventions, the *Anabæna* example can be depicted as shown in Figure 5.

We assume that IRN *variables* range over finite sets of values and may be updated using *variable update functions*. For each *global* or *local* variable *var*, there is a unique update function that computes the new value of *var* and whose allowed parameters are defined consistently with the constraints given in Figure 4 (*e.g.*, a global variable update may take as parameters only global variables or global measures as shown in the corresponding row). As for regulatory networks, x_{var} denotes the current value of variable *var*. By convention, the same name is used for the variable and its update function. For example, in *Anabæna* modeling, global variable *Nitro* can be specified by:

$$Nitro(x_{Nbr}) \in \{0, \ldots, n\}$$

for a given $n > 0$ and is defined according to Figure 3.

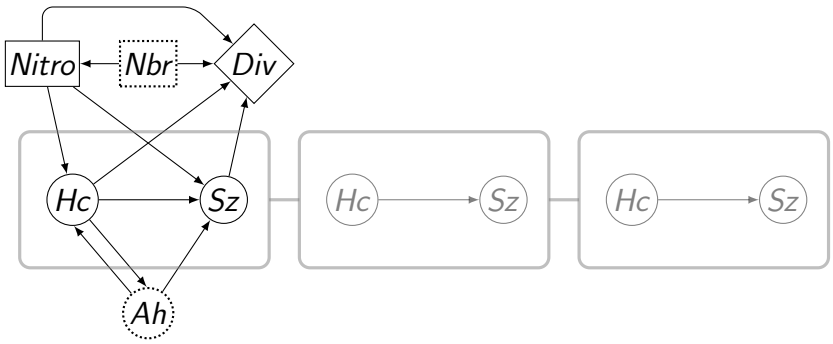

Figure 5: In black: graphical representation of the IRN specification of *Anabæna*; in the upper part, global variable *Nitro*, global measure *Nbr* and graph update *Div*; in the middle part, local variables *Hc* and *Sz*; in the lower part, local measure *Ah*. In gray, a view of a linear graph C with three vertexes with the corresponding local variables.

A local variable has a value in each module i in C, which is computed by an update function with the same name (common to all modules) that may take parameters as specified by Figure 4. For example, in *Anabæna* modeling, local variables *Hc* and *Sz* may be specified by giving their update functions:

$$\mathit{Hc}(x_{\mathit{Hc}i}, x_{\mathit{Nitro}}, x_{\mathit{Ah}i}) \in \{\mathsf{vegetative}, \mathsf{undetermined}, \mathsf{heterocyst}\}$$
$$\mathit{Sz}(x_{\mathit{Sz}i}, x_{\mathit{Nitro}}, x_{\mathit{Ah}i}) \in \{\mathsf{small}, \mathsf{big}\}$$

where $x_{\mathit{Ah}i}$ is the value of the local measure *Ah* computed for module i. They are defined according to Figure 3. The notation $x_{\mathit{lvar}i}$ is also used to denote the value of the local variable *lvar* in module i.

Measures are obtained from observations of the topological collection C. We distinguish *global measures* that are obtained by observing C globally, and *local measures* that are obtained by observing a module i and its neighborhood in C.

A global measure is defined by a function (with the same name) that returns a value in a finite set, taking as parameters C and the binding λ for the global variables, and possibly other parameters as allowed by Figure 4. For example, in the *Anabæna* modeling, global measure *Nbr* may be specified as:

$$\mathit{Nbr}(C, \lambda) \in \{0, \ldots, \mathit{Nbr}_m\}$$

returning the number of vertexes in C, bounded by Nbr_m.

A local measure *lm* is defined using a function with the same name that returns a value from a finite set, taking as parameters the values of *lvar* in

all the neighbors of i for each local variable *lvar* it depends on, and possibly other parameters as allowed by Figure 4. These values of each such *lvar*, denoted by $\overline{x_{\mathit{lvar}@i}}$, are collected as a multiset of pairs $(\ell_j, x_{\mathit{lvar}\,j})$, where ℓ_j is the label of the edge between i and j in C. In the *Anabæna* example local measure *Ah* may be specified as:

$$\mathit{Ah}(\overline{x_{\mathit{Hc}@i}}) \in \{\mathsf{true}, \mathsf{false}\}$$

returning true if there is at least one heterocyst in the neighborhood of i, and false otherwise.

A *graph update* is a function that takes as parameters a topological collection C, a binding λ for global variables, and a module identifier i. It either returns an empty set when the application conditions have not been met, or computes a set of new collections $\{C_1, \ldots, C_n\}$ $(n > 0)$. We assume that, for each C_k, $\mathit{vert}(C) \setminus \{i\} \subseteq \mathit{vert}(C_k)$ and all the modules in $\mathit{vert}(C) \setminus \{i\}$ are unchanged, *i.e.*, that the transformation is local to i. Notice that this does not forbid to make changes on the edges and their labellings, for instance, a cell migration will change the edges. In terms of topological collections, a graph update corresponds to a set of transformations. In the *Anabæna* example, graph update $\mathit{Div}(C, i)$ returns an empty set of graphs if the conditions for division are not met, or a singleton graph where the cell i has been replaced by two identical cells with $\mathit{Sz} = \mathsf{small}$.

3.3.2. Dynamics

A *state* of an IRN is represented by a pair (λ, C) where λ is a binding assigning values to the global variables, and C is a topological collection that records the graph structure and the value of the local variables as bindings attached to the vertexes. In other words, we have $x_{\mathit{gvar}} = \lambda(\mathit{gvar})$ for a global variable *gvar* and $x_{\mathit{lvar}\,i} = C(i)(\mathit{lvar})$ for a local variable *lvar* and a vertex i.

Schematically, a state (λ, C) of an IRN may evolve to another one (λ', C') in one of the following manners, if the corresponding application conditions are met and if $(\lambda', C') \neq (\lambda, C)$:

- by applying a graph update *gup* to the current topological collection for a module i: the resulting topological collection is C' returned by $\mathit{gup}(C, i)$, and the global variables are unchanged $\lambda' \stackrel{\text{df}}{=} \lambda$;
- by applying a global variable update: the resulting topological collection is unchanged $C' \stackrel{\text{df}}{=} C$, and binding λ' is λ updated for some global variable *gvar* such that $\lambda'(\mathit{gvar}) \stackrel{\text{df}}{=} \mathit{gvar}(\cdots)$;

- by applying a local variable update on a module i of C: the resulting topological collection C' is C where lvar at module i has been updated, *i.e.*, $C'(i)(\mathit{lvar}) \stackrel{\text{df}}{=} \mathit{lvar}(\cdots)$ and $C'(i')(\mathsf{v}) \stackrel{\text{df}}{=} C(i')(\mathsf{v})$ for every $i' \neq i$ and every $\mathsf{v} \neq \mathit{lvar}$, and the global variables are unchanged $\lambda' \stackrel{\text{df}}{=} \lambda$.

Given an initial state (λ_0, C_0), and the evolution rules defined above, one may build a transition system describing the dynamics of the IRN.

Implementation of the dynamics for Anabæna. In the *Anabæna* example, graph update Div may be implemented as the following MGS transformation on C:

$$
\begin{array}{l}
c_l, \quad \big(\{\ \mathsf{Sz} = \mathsf{big},\ \mathsf{Hc} = k\ \}\ \text{as}\ c\big), \quad c_r \\
/\ (\hat{}c = i) \ \ \wedge \ \ (\mathit{Nbr}(C,\lambda) < \mathit{Nbr}_m) \ \ \wedge \ \ (k \neq \mathsf{heterocyst}) \\
\quad \wedge \ \big(\lambda(\mathit{Nitro}) > 0 \ \vee \ c_l(\mathsf{Hc}) = \mathsf{heterocyst} \ \vee \ c_r(\mathsf{Hc}) = \mathsf{heterocyst}\big) \\
\rightarrow \ c_l,\ \mathrm{c}+\{\ \mathsf{Sz}{=}\mathsf{small}\ \},\ \mathrm{c}+\{\ \mathsf{Sz}{=}\mathsf{small}\ \},\ c_r
\end{array}
$$

where c_l and c_r match the left and the right neighbors of the cell c. The condition $\hat{}c = i$ holds when the vertex associated to cell c is i. The rest of the guard checks that the condition to divide are met (see section 3.2). Recall that, *e.g.*, c_l used in the rhs of the rule denotes the labeling of vertex $\hat{}c_l$, which is a binding that records the local variables of the corresponding module. So, for example, $c_l(\mathsf{Hc})$ is the value of Hc in module c_l. Two additional exclusive rules are defined similarly to handle the case of a cell at one end of the filament.

The value of a global or a local measure is obtained by invoking the corresponding function with the appropriate parameters. For example, Nbr is computed as:

$$\mathit{Nbr}(C, \lambda) = \mathit{size}(C)$$

where the function *size* in MGS returns the number of vertexes in the topological collection.

A local measure lm, needs to read the current values of each involved local variable lvar in all the neighbors of module i. For example, local measure Ah for a module i can be computed in MGS using function $\mathit{NeighborFold}(i, f, \mathit{init})$. This is a higher-order function that iterates a binary reduction function f over the labels of the neighbors of i to build up a return value. The argument *init* is used to initialize the accumulator. In our example, we have:

$$\mathit{Ah}(C, \lambda, i) = \mathit{NeighborFold}(i, (\backslash x, \mathit{acc}.\mathit{acc} \vee (x(\mathsf{Hc}) = \mathsf{heterocyst})), \mathsf{false})$$

where $\backslash x_1, \cdots, x_k.expr$ is the MGS notation for a lambda abstraction.

The update of a global variable *gvar* corresponds to a call to the corresponding update function with the appropriate parameters computing a new value for *gvar* belonging to $dom(\mathit{gvar})$. For example, the update function of global variable *Nitro* is defined as follows:

$$\mathit{Nitro}(x_{\mathit{Nbr}}) = \text{ if } (x_{\mathit{Nitro}} > 0) \wedge (x_{\mathit{Nbr}} \geq \mathit{Nbr}_N) \text{ then } x_{\mathit{Nitro}} - 1 \text{ else } x_{\mathit{Nitro}}$$

The update of a local variable *lvar* in a module i is defined similarly and corresponds to a call to the corresponding update function with the appropriate parameters, computing $x_{\mathit{lvar}\,i}$, a new value for *lvar* in i, which belongs to $dom(\mathit{lvar})$. The actual updating is applied through a transformation with a single rule. For example, in the *Anabæna* modeling, the application of the update function for the size of a cell is given by

$$c \,/\, (\hat{}c = i) \rightarrow c + \{\mathit{Sz} = \mathit{Sz}(c(\mathit{Sz}), \lambda(\mathit{Nitro}), \mathit{Ah}(C, \lambda, i))\}$$

where function *Sz* is specified in Figure 3.

3.4. Properties of IRNs

3.4.1. Conservative extension of existing formalisms

As we have shown in our incremental presentation, IRNs form a *conservative extension* of various formalisms. First, the generalized logical formalisms [43] may be recovered by using a single module with only local variables whose update functions are compatible with logical regulatory networks (in particular, they enforce variations by steps of ± 1). The extension with modules from [8] can then be recovered by using several modules with only local variables, local measures to implement so-called integration functions (used to aggregate the values of local variables in neighbor modules), and labeling the edges of the graph by real numbers in the segment $[0; 1]$ (an abstract notion of distance). The dynamics in [8] is given by colored Petri nets executions while IRNs are executed as transition systems. There is an obvious correspondence between both because every transition of the Petri net implements exactly one update function. Finally, we have seen at the beginning of section 3 how the current work is derived from our own extension of [8] proposed in [15].

3.4.2. D0L systems encoding

IRNs can easily encode some formalisms proposed to model biological development [29], in particular *D0L systems*. A *D0L system* is a triple

$G = (\Sigma, h, \omega)$ where Σ is an alphabet, h is a finite substitution on Σ (into Σ^*) and ω, the initial word, is an element of Σ^+. Letter "D" stands for deterministic (the derivation sequence is unique) and the numerical argument of the L system gives the number of interactions in the rewriting process: a 0L system is a context free L system. For D0L system, there exists at most a single production rule for each element a of Σ given by $a \to h(a)$. (We assume L systems such that the length of $h(a)$ for all a is strictly greater than 1.) The translation of the production rules in topological rewriting rules is straightforward: the production rules $a \to a_1 \cdots a_k$ in the L system translates into the equivalent MGS rules: $\mathsf{a} \to \mathsf{a_1}, \cdots, \mathsf{a_k}$ for each symbol $a \in \Sigma$. However, a maximal parallel rewriting strategy must be used to respect the synchronous evolution of L systems, which we will emulate.

If each symbol is represented as the value of a local variable, this approach does not translate directly into an IRN, because the IRN evolutions of modules are asynchronous. Thus, we decide to manage the application explicitly, using a flag which indicate where a production can take place. Our approach is to apply the production sequentially, letter by letter, from left to right, starting from the first letter.

More precisely, we consider an oriented ring topology. A module is composed of three local variables: S with values in Σ, Start with a Boolean value indicating the beginning of the word in the ring topology (there is exactly one module with a variable Start set to true) and $\mathit{Flag} \in \{\mathsf{quiet}, \mathsf{done}, \mathsf{ready}, \mathsf{go}\}$ used to trigger the application of a production.

The application of a production is translated into a graph update function consisting of a rule for each production $\mathsf{a} \to \mathsf{a_1} \ldots \mathsf{a_k}$ of the D0L system:

$$
\begin{aligned}
\{\mathit{S} = \mathsf{a}, \mathit{Start} = s, \mathit{Flag} = \mathsf{go}\} \quad \to \quad & \{\mathit{S} = \mathsf{a_1}, \mathit{Start} = s, \mathit{Flag} = \mathsf{quiet}\}, \\
& \{\mathit{S} = \mathsf{a_2}, \mathit{Start} = \mathsf{false}, \mathit{Flag} = \mathsf{quiet}\}, \\
& \quad \ldots, \\
& \{\mathit{S} = \mathsf{a_{k-1}}, \mathit{Start} = \mathsf{false}, \mathit{Flag} = \mathsf{quiet}\}, \\
& \{\mathit{S} = \mathsf{a_k}, \mathit{Start} = \mathsf{false}, \mathit{Flag} = \mathsf{done}\}
\end{aligned}
$$

We assume that there is exactly one module with Flag equal to go in the initial state and this module has Start set to true. When the graph update function applies, it substitutes the module with a go flag by several new modules.

$$
\begin{aligned}
&a_g, b_q, c_q, d_q, e_q, f_q, g_q, h_q \\
&\xrightarrow{a_g} A_q, A_d, b_q, c_q, d_q, e_q, f_q, g_q, h_q \\
&\xrightarrow{b_q} A_q, A_d, b_r, c_q, d_q, e_q, f_q, g_q, h_q \\
&\xrightarrow{A_d} A_q, A_q, b_r, c_q, d_q, e_q, f_q, g_q, h_q \\
&\xrightarrow{b_r} A_q, A_q, b_g, c_q, d_q, e_q, f_q, g_q, h_q \\
&\xrightarrow{b_g} \cdots \\
&\xrightarrow{h_g} A_q, A_q, B_q, B_q, C_q, C_q, D_q, D_q, E_q, E_q, F_q, F_q, G_q, G_q, H_q, H_d \\
&\xrightarrow{A_r} A_r, A_q, B_q, B_q, C_q, C_q, D_q, D_q, E_q, E_q, F_q, F_q, G_q, G_q, H_q, H_d \\
&\xrightarrow{H_d} A_r, A_q, B_q, B_q, C_q, C_q, D_q, D_q, E_q, E_q, F_q, F_q, G_q, G_q, H_q, H_q \\
&\xrightarrow{A_r} A_g, A_q, B_q, B_q, C_q, C_q, D_q, D_q, E_q, E_q, F_q, F_q, G_q, G_q, H_q, H_q
\end{aligned}
$$

Figure 6: A complete evolution of an IRN encoding a D0L system with rules such that each letter is replaced by two copies in upper case. The value of the flag is given by the subscript, the left-most cell is the one starting the word and the ring is left-right oriented.

The local variable *Flag* evolves following the update function below:

$$
\mathit{Flag}(x_{\mathit{Prev}\,i}, x_{\mathit{Next}\,i}) \stackrel{\text{df}}{=} \begin{cases} \mathsf{quiet} & \text{if } x_{\mathit{Flag}} = \mathsf{done} \wedge x_{\mathit{Next}\,i} = \mathsf{ready} \\ \mathsf{ready} & \text{if } x_{\mathit{Flag}} = \mathsf{quiet} \wedge x_{\mathit{Prev}\,i} = \mathsf{done} \\ \mathsf{go} & \text{if } x_{\mathit{Flag}} = \mathsf{ready} \wedge x_{\mathit{Prev}\,i} = \mathsf{quiet} \\ x_{\mathit{Flag}} & \text{for the other cases} \end{cases}
$$

In this function, local measures *Prev* and *Next* give the value of *Flag* at the previous and the next module (with respect to the ring orientation). Figure 6 illustrates the propagation of the flags.

The initial state of the system is composed of modules with variable *Start* set to false except for one module m, and with variable *Flag* set to quiet, except for the module m where it is equal to go. A word generated by the L system can be read as the values of S (in the direction of the ring), starting from the unique module m with *Start* set to true, when $x_{\mathit{Flag}\,m}$ is equal to go.

3.4.3. Further properties of IRNs

The IRN formalism enjoys also some other interesting properties and we want to stress three of them in particular. The first one is that the size of a model specification is independent from the number of cells in the system. This means that the description of the dynamics, through the various update

functions, is generic. What is remarkable is that this is not true if we simply unfold over the cells a logical formalism *à la* Thomas.

A second important property is that our framework is *polytypic*, that is, generic with respect to the topology of the underlying graph [25]. This property enables for example to use the same model on NEWS grids or on hexagonal grids (see below, section 4.2). Polytypism is achieved in our case through the use of the polytypic neighborhood operator (the coma operator described in section 2.3.2) or iterators like *NeighborFold* (section 3.3.2).

Finally, thanks to syntactical restrictions and the definition of the dynamics, the transition system (state space) of an IRN is always finite. This enables various analysis techniques, in particular model-checking.

4. Applications

4.1. Anabæna *continued*

Thanks to the previous formalization, we can model-check various properties of the *Anabæna* dynamics. Figure 7 shows an explicit representation of the transition system for a given initial configuration. On such graphs, it can be checked, for example, that no reachable state has two neighbor heterocysts. In the instance depicted in Figure 7, one can see that nitrogen does not decrease before the initial cell has divided, which corresponds to the chosen parameter $\mathsf{Nbr}_N \stackrel{\text{df}}{=} 2$.

The transition system has been computed using our prototype that includes the set of MGS functions already presented. Because MGS is a simulation tool, it could compute only one trajectory among the possible ones (*i.e.*, one maximal path from the top-most node in figure 7). Such a state space is perfectly suitable to perform various model-checking approaches: for instance it is easy to check that no reachable state has two neighbor heterocysts. More elaborated properties may be expressed using a temporal logic and checked automatically using appropriate tools.

To compute the whole state space, we have used an external program that drives MGS to perform the actual computation while the program is only responsible for building the transition system ensuring that all the possible executions have been explored. The coupling between MGS and the exploration program is currently made exploiting the capability of MGS to be used in interactive mode: the driver program sends commands to the MGS shell and retrieves the corresponding outputs. This way of coupling works well on small examples like those considered in this paper but it has

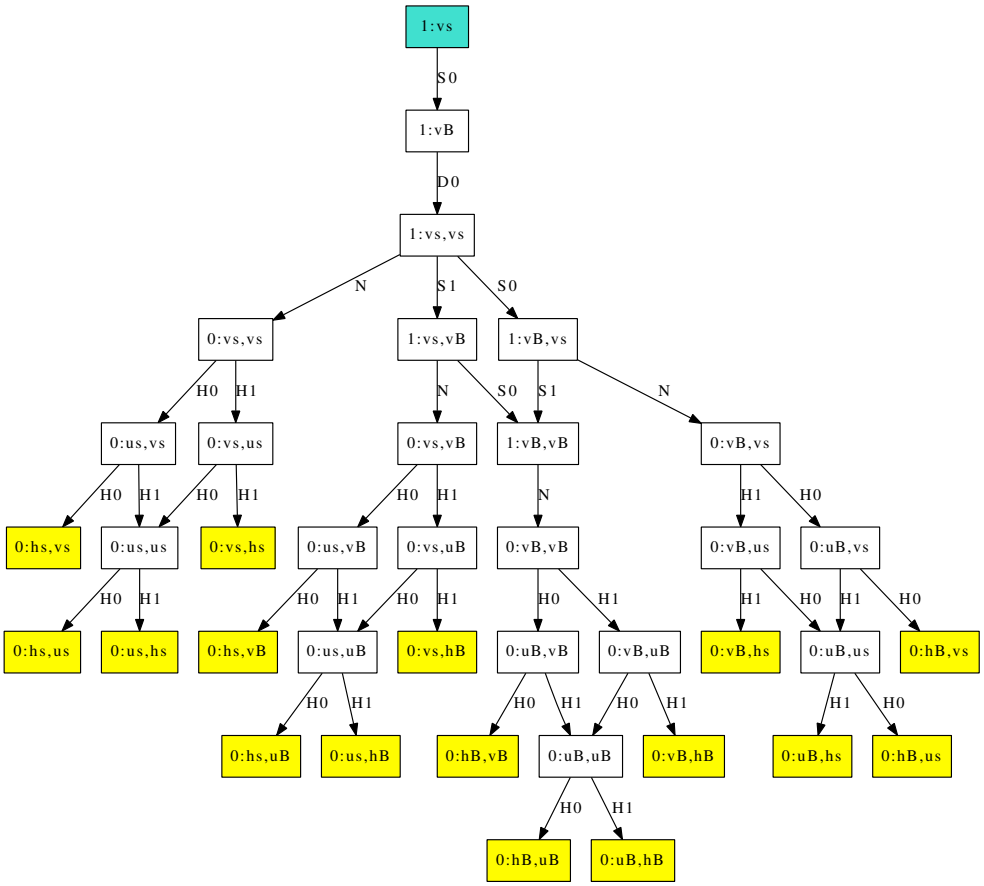

Figure 7: The transition system generated from the initial state corresponding to a single small vegetative cell and $x_{Nitro} \stackrel{\text{df}}{=} 1$, with both parameters Nbr_m and Nbr_N set to 2. Each state is depicted as a rectangle whose label gives first x_{Nitro} then, for each module, the value of each local variable Hc and Sz encoded by the initial character of the corresponding symbol. For example, 0:hB,vs stands for an IRN state with $\lambda \stackrel{\text{df}}{=} \{Nitro = 0\}$ and $C \stackrel{\text{df}}{=} \{Hc = \text{heterocyst}, Sz = \text{big}\}, \{Hc = \text{vegetative}, Sz = \text{small}\}$ (two vertexes linked by comma operator). There is an edge between two nodes iff a corresponding transition exists. Edge labels are either Hi, Si or N, corresponding respectively to an evolution of Hc, Sz in module i, or $Nitro$ globally. Leaf nodes in this directed acyclic graph (there is no periodic behavior) are stable states.

at least two important limitations: it is slow because its requires to convert data from/to text, and it is error prone because such a conversion is complex in general. A way to overcome this problem is to use for a future implementation a lower-level coupling through the internal API of MGS.

Nbr_m	Nbr_N	initial state	states	transitions	stable states
2	2	1:vs	39	42	16
2	2	2:vs	43	50	16
2	2	3:vs	47	58	16
3	2	1:vs	223	346	56
3	2	2:vs	235	378	56
3	2	3:vs	247	410	56
3	3	1:vs	191	286	48
3	3	2:vs	199	306	48
3	3	3:vs	207	326	48
5	4	3:vs	5375	12466	896
5	4	3:vs,vs	5373	12464	896
5	4	3:vs,vs,vs,vs	5361	12432	896

Table 1: State space size for the *Anabæna* example wrt various initial configurations.

To conclude about the *Anabæna* example, table 1 shows the evolution of the state space size with respect to the choice of various parameters. It shows in particular that the well known state space explosion phenomenon mainly depends on Nbr_m. This is not surprising: the number of states clearly grows exponentially with the number of cells, but parameter Nbr_N and the initial value of x_{Nitro} are global and do not really increase the combinatorial explosion (in particular, they have no influence on the stable states). Similarly, adding more cells to the initial state does not reduce significantly the state space when the chosen configuration is not close to a stable state.

4.2. Quorum sensing

The *Anabæna* example shows the interplay between the logical and the physical part of a system. However, it involves only a simple linear organization of the cells.

In this second example, we will sketch the use of topological collections to model a more elaborate geometry and the benefit of polytypism. We are interested in the sketch of a bacterial communication mechanism promoting collective behavior within a population: *quorum sensing*. This term refers to a type of decision-making process whereby the detection and the synthesis of small diffusing autoinducer molecules enable a single cell to coordinate its behavior with the rest of the bacterial population. This mechanism has

mainly been proposed to serve as a means to regulate gene expression with the local density of the cell population [47].

In our example, we assume that the sources of the signaling molecules are a specialized cell type within a tissue or a biofilm. The spatial organization of the tissue will be modelled by *Group Based Fields* or *GBFs* that generalize the idea of grids to several dimensions and several neighborhood structures.

We first present the notion of GBF then a simple model of quorum sensing.

4.2.1. Group based fields

GBF are topological collections whose vertexes are elements in an Abelian group [16]. The group is defined by a *finite presentation*:

$$G = \langle \mathsf{g}_1, \ldots, \mathsf{g}_n; w_1 = 0, \ldots w_n = 0 \rangle$$

where $G_g = \{\mathsf{g}_1, \ldots, \mathsf{g}_n\}$ is a set of generators together with some constraints w_j on their combinations: w_j is a group element which equates to zero (we use an additive notation and 0 denotes the identity element of the group). The graph underlying the GBF is the Cayley graph of the finite presentation: the set of vertexes $\mathbb{V}$ is composed of the elements of the group G and there is an edge labeled by $\mathsf{g}_k \in G_g$ between the vertexes h and h' iff $h + \mathsf{g}_k = h'$.

For instance, in order to define a square grid, also called a NEWS grid (north, east, west, south), we may use two generators e (east) and n (north). This is illustrated in the left part of Figure 8. Similarly, a hexagonal grid can be defined by means of three generators n, e and nw (north-west) and a constraint $\mathsf{n} - \mathsf{nw} - \mathsf{e} = 0$, as illustrated in the right part of Figure 8. As shown by the dashed path, we have $2 \cdot \mathsf{n} + \mathsf{e} = 2 \cdot \mathsf{e} + \mathsf{n} + \mathsf{nw}$, which can be also checked in an algebraic way, by substituting nw with $\mathsf{n} - \mathsf{e}$ in this equality as allowed by the constraint. Notice that this grid is called hexagonal rather than triangular because the grid can be paved by hexagonal cells placed on the vertexes.

The GBF structure is thus adequate to define the arrangement on a grid, in any number of dimensions. In such grids, a distance can be naturally defined as the minimum number of steps in order to reach one point from the other (this is the approach of *geometric group theory*). For instance, in the hexagonal grid of Figure 8, points at e and n are at distance 1 because only one step in direction nw is required to reach the latter from the former; similarly, points n and $2 \cdot \mathsf{n} + \mathsf{e}$ are at distance 2.

The main drawback with grids is that inserting new elements is not possible: there must exist a "hole", *i.e.*, an unallocated vertex, to place the cell

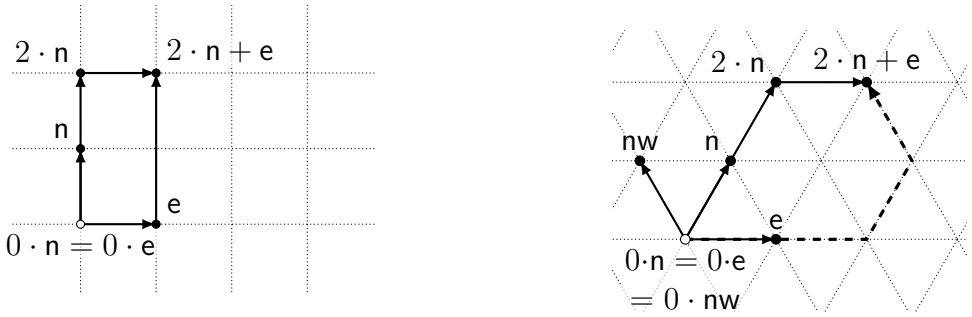

Figure 8: Left: a GBF defining a square grid, with two generators e and n. Right: a GBF defining a hexagonal grid with three generators e, n and nw, and a constraint n − nw = e.

to be inserted. However, the simplicity of grids is well adapted to model, for instance, accretive growth that occurs at the borders of a tissue [19].

4.2.2. A simple discrete quorum-sensing process

The purpose of the toy model we develop here is to illustrate the spatial representation capabilities of IRNs and not to develop a realistic or a relevant model of quorum sensing. Thus, we simplify this mechanism as follows: we assume that each cell is able to sense a signaling molecule (inducer) in its immediate environment (*e.g.*, the inducer can bind to a membrane receptor and induces the transcription of some genes) and then activate the synthesis of the inducer itself.

The cells in a tissue are represented by modules organized in a GBF. Each cell is characterized by a local variable M ranging over $\{0, \ldots, M\}$ and representing a discrete level of the signaling molecules produced by the cell. The concentration M evolves in a cell accordingly of the local concentration computed as a local measure D. To simplify, we assume that D simply computes the average value of the concentrations in the neighbors, which can be implemented in MGS as:

$$D(C, \lambda, i) = \mathrm{round}\big(\mathit{NeighborFold}(i, +, 0) / \mathit{NeighborFold}(i, \mathit{succ}, 0)\big)$$

where *succ* is the successor function on integers. Then, the variable update function for M simply computes the average between the current value of M in the cell and the value observed through D in the neighborhood (*i.e.*, the cell evolves toward the concentration sensed in the neighborhood). Finally,

we assume that cells whose initial concentration is M are sources so that M does not change for them.

We choose initial states in which all the non-source cells have M set to 0. This results in stable states showing gradients of concentrations decreasing from the source cells. The gradient is the consequence of the rounding to integers that forbid a progressive convergence toward the concentration of source cells (as it would happen in a continuous setting) and creates a decreasing of the sensed average values as cells are farther from the sources.

Two examples are represented in Figure 9, one uses the NEWS grid and the other the hexagonal grid. This shows the polytipic features of IRNs: the specification of the measure and of the update functions remains the same in the two examples (including their MGS implementations), only the type of the underlying topological collection changed. Figure 10 shows the transition system of a quorum sensing process. One can see that the chosen topology is crucial to determine the system evolution and the final pattern.

5. Related and future works

In this paper, we advocate the need of a modeling framework suitable to represent and study the regulations in multi-cellular systems, taking into account the spatial relationships between the cells as well as the spatial transformations resulting from cells divisions, migrations, or apoptosis. Discrete algebraic formalisms like P systems, process algebras or Petri nets are very relevant because automated tools can be used to help both the modeling and the systematic analysis of system behaviors (cf. [1] for the model-checking of P systems). Such formalisms may take spatial relationships into account. For instance, classical membrane systems rely on membranes inclusion to abstract the spatial organization of cellular processes. However, the limitations of this structure has been recognized [17] and leads to the development of several variants: tissue P systems [45], population P systems [4], etc. Some process algebras (*e.g.*, used to study mobility or variants of π-calculus used for biological modelling) rely also on a notion of localization but often the spatial relationships are not explicitly exposed (the algebra of locations is encoded into identifiers) or too limited (nesting structures). [3, 6, 26] are among the rare works where geometry (especially a notion of distance) is embedded in a process algebra to deal with the dynamic spatial arrangement of cells through simulation. These frameworks focus on the articulation of

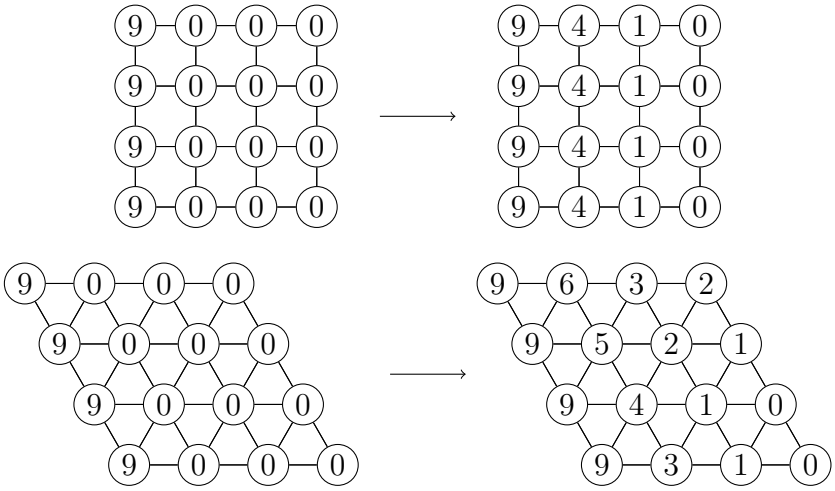

Figure 9: Each row shows the initial and final state of a quorum sensing process. Only the declaration of the underlying spatial representation changes between the two rows: exactly the same variables, update functions and implementations have been used.

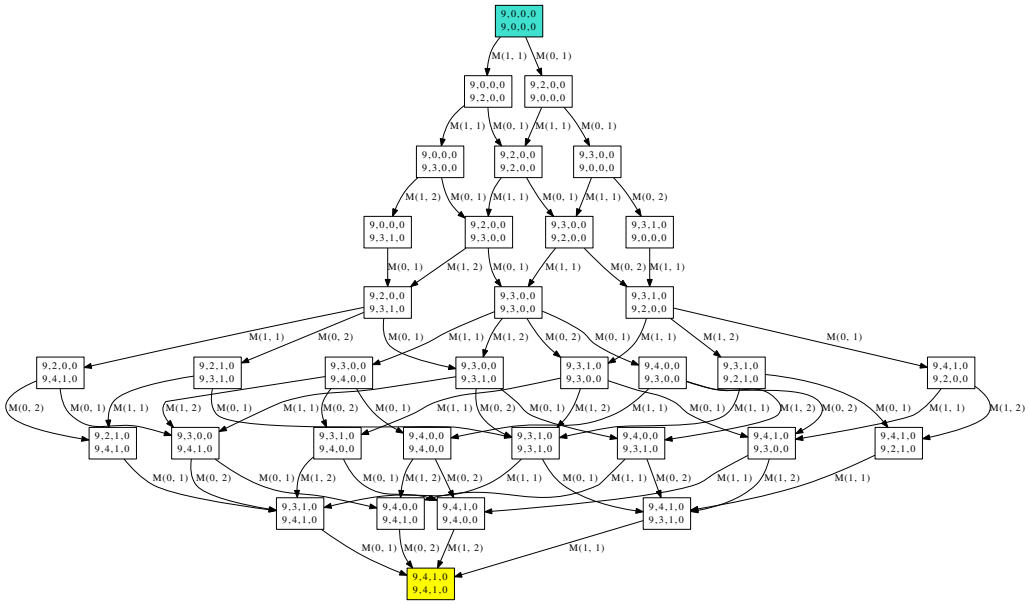

Figure 10: The transition system of a quorum sensing process for two rows of cells on a square grid.

local coordinate frames but the affine maps do not allow enumeration of the state spaces.

The IRN framework presented in this paper, based on the well-known formalism of logical regulatory networks combined with spatial information, may be enriched and extended in several directions.

First, the current definition of IRNs only supports systems having a unique kind of modules. However, it is due to a deliberate presentation decision allowing us to focus on essential IRN concepts using simple notations. Natural extensions to many different kinds of modules may easily be provided and are already handled by our prototype implementation which does not have any such limitations. Future works will provide a generalized formal definition of the framework presented in the paper. We also intend to run case studies in order to assess the relevance of our proposal: in particular we shall consider models of *Drosophilia* embryo as in [8] and extend them with cells divisions.

Second, concerning spatial transformations, we focus here on labeled graphs. However, our mathematical description is not based on the usual graph morphisms and pushouts like in [13] but is inspired by the approach of J.-C. Raoult [32] where graph rewriting based on a (multi-)set point of view is developed. The proposed model is close to term rewriting modulo associativity and commutativity (where the left hand side of a rule is removed and the right hand side is added). This kind of approach also allows to extend results from term rewriting to topological rewriting (as we did for termination in [21]). Note that the notions of topological collection and topological rewriting are more general and may handle higher dimensional objects, a feature relevant in a lot of application areas [44].

Another direction of future research consists in relaxing some constraints concerning the dynamics definition; for example, to allow alternative update strategies or infinite state spaces. Concerning the former, the current framework defines the dynamics of the system using an asynchronous strategy. This approach is relevant, *e.g.*, for regulation networks. But we have seen in section 3.4.2 an example that could be more easily expressed using a synchronous maximal parallel update strategy. The design space of update strategy is largely open, from asynchronous to synchronous and from deterministic to non-deterministic ones. For the latter research direction, the actual IRN specification restrictions ensure finiteness of the state space, but may become artificial in practice. Some of these restrictions may be relaxed by resorting to abstraction and specific reduction techniques, like those in [27, 28], which

originally have been designed for systems with dynamic process creation.

Finally, we intend to study an alternative approach for spatial information in order to address systems such as blood-cells populations. The idea is to implement the spatial relation as a purely stochastic relation reflecting the idea that such cells are in constant movement but may meet occasionally.

Acknowledgements. The authors thank F. Delaplace at the University of Évry and the organizers and participants of MeCBIC'2010 for fruitful discussions and for their valuable comments. This work is partially supported by the ANR research projects AutoChem, Calamar and SynBioTIC.

References

[1] O. Andrei, G. Ciobanu, and D. Lucanu. A rewriting logic framework for operational semantics of membrane systems. *Theor. Comput. Sci.*, 373(3):163–181, 2007.

[2] P. Barbier de Reuille, I. Bohn-Courseau, K. Ljung, H. Morin, N. Carraro, C. Godin, and J. Traas. Computer simulations reveal properties of the cell-cell signaling network at the shoot apex in Arabidopsis. *Proceedings of the National Academy of Sciences of the United States of America*, 103(5):1627–1632, 2006.

[3] E. Bartocci, F. Corradini, M.R. Di Berardini, E. Merelli, and L. Tesei. Shape calculus. a spatial mobile calculus for 3d shapes. *Scientific Annals of Computer Science*, 20:1–31, 2010.

[4] F. Bernardini and M. Gheorghe. Population P systems. *Journal of Universal Computer Science*, 10(5):509–539, 2004.

[5] Arne T. Bittig and Adelinde M. Uhrmacher. Spatial modeling in cell biology at multiple levels. In *Winter Simulation Conference*, pages 608–619, 2010.

[6] L. Cardelli and P. Gardner. Processes in space. *Programs, Proofs, Processes*, pages 78–87, 2010.

[7] C. Chaouiya, H. Klaudel, and F. Pommereau. A Petri net based framework for a qualitative modelling of regulatory networks encompassing inter-cellular signalling. In A. Navarro and A. Oliveira, editors, *JB'09*, pages 1–5, 2009.

[8] C. Chaouiya, H. Klaudel, and F. Pommereau. *A modular, qualitative modelling of regulatory networks using Petri nets*, chapter 12 of *Modeling in Systems Biology – the Petri Net Approach*. Springer, 2010.

[9] C. Chaouiya, A. Naldi, E. Remy, and D. Thieffry. Petri net representation of multi-valued logical regulatory graphs. *Natural Computing*, 10:727–750, 2011.

[10] C. Chaouiya, E. Remy, and D. Thieffry. Qualitative Petri net modelling of genetic networks. *Trans Comp Syst Biol*, VI(4220):95–112, 2006.

[11] E. Coen, A.G. Rolland-Lagan, M. Matthews, J.A. Bangham, and P. Prusinkiewicz. The genetics of geometry. *Proceedings of the National Academy of Sciences of the United States of America*, 101(14):4728, 2004.

[12] F. Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, 1970.

[13] H. Ehrig, M. Pfender, and H. J. Schneider. Graph grammars: An algebraic approach. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 1973.

[14] J.-L. Giavitto. Topological collections, transformations and their application to the modeling and the simulation of dynamical systems. In *14th International Conference on Rewriting Techniques and Applications (RTA 2003)*, LNCS 2706, pages 208–233. Springer, 2003.

[15] J.-L. Giavitto, H. Klaudel, and F. Pommereau. Qualitative modelling and analysis of regulations in multi-cellular systems using Petri nets and topological collections. In *Proceedings Fourth Workshop on Workshop on Membrane Computing and Biologically Inspired Process Calculi (MeCBIC 2010)*, volume 40. Electronic Proceedings in Theoretical Computer Science (EPTCS), 2010. ArXiv e-prints 1011.0498.

[16] J.-L. Giavitto and O. Michel. 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*, pages 150–161, New York, NY, USA, 2001. ACM Press.

[17] J.-L. Giavitto and O. Michel. The topological structures of membrane computing. *Fundamenta Informaticae*, 49:107–129, 2002.

[18] J.-L. Giavitto and O. Michel. Modeling the topological organization of cellular processes. *BioSystems*, 70:149–163, 2003.

[19] J.-L. Giavitto, O. Michel, and J. Cohen. Accretive rules in cayley p systems. In *Selected paper from the International Workshop on Membrane Computing (WMC-CdeA 2002)*, LNCS 2597, pages 319–338. Springer, 2002.

[20] J.-L. Giavitto, O. Michel, and J. Cohen. Pattern-matching and rewriting rules for group indexed data structures. *SIGPLAN Not.*, 37:76–87, 2002. Special issue on selected papers from the 2002 PLI workshops.

[21] J.-L. Giavitto, O. Michel, and A. Spicher. Spatial organization of the chemical paradigm and the specification of autonomic systems. In *Software-Intensive Systems and New Computing Paradigms*, volume 5380 of *LNCS*, pages 235–254. Springer, 2008.

[22] J.-L. Giavitto and A. Spicher. Topological rewriting and the geometrization of programming. *Physica D: Nonlinear Phenomena*, 237(9):1302–1314, 2008.

[23] M. Hammel and P. Prusinkiewicz. Visualization of developmental processes by extrusion in space-time. In *Graphics Interface*, pages 246–257, 1996.

[24] M. Hammel and P. Prusinkiewicz. Visualization of developmental processes by extrusion in space-time. In *Graphics Interface*, pages 246–258, 1996.

[25] J. Jeuring and P. Jansson. Polytypic programming. In *Advanced Functional Programming*, volume 1129 of *LNCS*, pages 68–114. Springer, 1996.

[26] M. John, R. Ewald, and A.M. Uhrmacher. A spatial extension to the π calculus. *Electronic Notes in Theoretical Computer Science*, 194(3):133–148, 2008.

[27] H. Klaudel, M. Koutny, E. Pelz, and F. Pommereau. An approach to state space reduction for systems with dynamic process creation. In *ISCIS'09*, IEEE digital library. IEEE, 2009.

[28] H. Klaudel, M. Koutny, E. Pelz, and F. Pommereau. State space reduction for dynamic process creation. *SACS, "Alexandru Ioan Cuza" University of Iaşi*, 20, 2010.

[29] A. Lindenmayer. Mathematical models for cellular interaction in development, Parts I and II. *Journal of Theoretical Biology*, 18:280–315, 1968.

[30] G. J. Mitchinson and M. Wilcox. Rule governing cell division in anaeba. *Nature*, 239:110–11, 1972.

[31] J. Munkres. *Elements of Algebraic Topology*. Addison-Wesley, 1984.

[32] J.-C. Raoult and F. Voisin. Set-theoretic graph rewriting. In *Proceedings of the International Workshop on Graph Transformations in Computer Science*, pages 312–325, London, UK, 1994. Springer-Verlag.

[33] Didier Rémy. Syntactic theories and the algebra of record terms. Technical Report 1869, INRIA-Rocquencourt, BP 105, F-78 153 Le Chesnay Cedex, 1992.

[34] G. Rozenberg and A. Salomaa, editors. *Lindenmayer Systems: Impacts on Theoretical Computer Science, Computer Graphics and Developmental Biology*. Springer Verlag, Berlin, Heidelberg, New York, 1992.

[35] J. Saez-Rodriguez, L. Simeoni, J. A. Lindquist, A. Hemenway, U. Bommhardt, B. Arndt, U-U. Haus, R. Weismantel, E. Gilles, S. Klamt, and B. Schraven. A logical model provides insights into T cell receptor signaling. *PLoS Comput Biol*, 3(8):e163, 2007.

[36] L. Sánchez, C. Chaouiya, and D. Thieffry. Segmenting the fly embryo: a logical analysis of the segment polarity cross-regulatory module. *Int. J. Dev. Biol.*, 52(8):1059–75, 2008.

[37] A. Spicher, O. Michel, M. Cieslak, J.-L. Giavitto, and P. Prusinkiewicz. Stochastic p systems and the simulation of biochemical processes with dynamic compartments. *BioSystems*, 91(3):458–472, 2008.

[38] A. Spicher, O. Michel, and J.-L. Giavitto. Declarative mesh subdivision using topological rewriting in mgs. In *Int. Conf. on Graph Transformations (ICGT) 2010*, volume 6372 of *LNCS*, pages 298–313, 2010.

[39] A. Spicher, O. Michel, and J.-L. Giavitto. *Understanding the Dynamics of Biological Systems: Lessons Learned from Integrative Systems Biology*, chapter Interaction-Based Simulations for Integrative Spatial Systems Biology. Springer Verlag, 2011.

[40] K. Takahashi, S. N. V. Arjunan, and M. Tomita. Space in systems biology of signaling pathways - towards intracellular molecular crowding in silico. *FEBS Letters*, 579(8):1783–1788, 2005.

[41] R. Thomas. Boolean formalization of genetic control circuits. *J. Theor. Biol.*, 42:563–85, 1973.

[42] R. Thomas. Regulatory networks seen as asynchronous automata: A logical description. *J. Theor. Biol.*, 153:1–23, 1991.

[43] R. Thomas, D. Thieffry, and M. Kaufman. Dynamical behaviour of biological regulatory networks–I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state. *Bull. Math. Biol.*, 57:247–76, 1995.

[44] E. Tonti. On the mathematical structure of a large class of physicial theories. *Rendidiconti della Academia Nazionale dei Lincei*, 52(fasc. 1):48–56, 1972. Scienze fisiche, matematiche et naturali, Serie VIII.

[45] C.M. Vide, J. Pazos, G. Paun, and A.R. Patón. Tissue P systems. *Theoretical Computer Science*, 296:295–326, 2003.

[46] M. Wilcox, G. J. Mitchison, and R. J. Smith. Pattern formation in the blue-green alga, *Anabaena*. I. Basic mechanisms. *Journal of Cell Science*, 12:707–723, 1973.

[47] P. Williams. Quorum sensing, communication and cross-kingdom signalling in the bacterial world. *Microbiology*, 153(12):3923–3938, 2007.