




# A survey of gene regulatory networks modelling methods: from differential equations, to Boolean and qualitative bioinspired models

Roberto Barbuti<sup>1</sup> · Roberta Gori<sup>1</sup> · Paolo Milazzo<sup>1</sup>  · Lucia Nasti<sup>1</sup>

Received: 14 March 2020 / Accepted: 13 August 2020 / Published online: 8 September 2020  
© The Author(s) 2020

## Abstract

Gene Regulatory Networks (GRNs) represent the interactions among genes regulating the activation of specific cell functionalities, such as reception of (chemical) signals or reaction to environmental changes. Studying and understanding these processes is crucial: they are the fundamental mechanism at the basis of cell functioning, and many diseases are based on perturbations or malfunctioning of some gene regulation activities. In this paper, we provide an overview on computational approaches to GRN modelling and analysis. We start from the biological and quantitative modelling background notions, recalling differential equations and the Gillespie's algorithm. Then, we describe more in depth qualitative approaches such as Boolean networks and some computer science formalisms, including Petri nets, P systems and reaction systems. Our aim is to introduce the reader to the problem of GRN modelling and to guide her/him along the path that goes from classical quantitative methods, through qualitative methods based on Boolean network, up to some of the most relevant qualitative computational methods to understand the advantages and limitations of the different approaches.

**Keywords** Gene regulatory networks · Boolean networks · Petri nets · Reaction systems · Membrane systems

## 1 Introduction

*Gene Regulatory Networks (GRNs)* [81] are the mechanism that allows cells to react to environmental changes such as the availability of a new nutrient or the reception of a (chemical) signal from other cells. A cell activates a new function by starting synthesizing different proteins. Indeed, proteins are the actuators of cell functions and each protein plays a rather specific role. The synthesis of proteins is based on genes, through the transcription (synthesis of

RNA from DNA) and translation (synthesis of proteins from RNA) processes. The activation of a new cell function corresponds to the activation of the transcription and translation mechanisms. With a little simplification, each gene can be considered in *active* or *inactive* state depending on whether the corresponding protein is expressed (i.e. synthesized) or not. This allows mapping cell functionalities to specific configurations of genes activation.

Since each cell functionality is often associated to a large number of genes, its activation has to be properly coordinated. This is obtained through a distributed process in which genes mutually regulate their activation. Interactions among genes via proteins, in which each gene promotes (i.e. stimulates) or inhibits the activation of one or more other genes, can be described in terms of a network. In this kind of network, nodes represent genes and (oriented) connections (of different types) represent the influence that each gene has on each other. This qualitative way of describing gene regulation activities is the most common approach to the representation of GRNs. It is a simplification of a more complex and quantitative process which involves RNA and protein synthesis, chemical interactions, and so on, and that could be described in terms of Ordinary Differential

---

Roberto Barbuti, Roberta Gori, Paolo Milazzo and Lucia Nasti contributed equally.

✉ Paolo Milazzo  
milazzo@di.unipi.it

Roberto Barbuti  
barbuti@di.unipi.it

Roberta Gori  
gori@di.unipi.it

Lucia Nasti  
lucia.nasti@di.unipi.it

<sup>1</sup> Dipartimento di Informatica, Università di Pisa, Pisa, Italy

Equations (ODEs) defined in accordance with standard chemical kinetic laws. Quantitative models often require too much (unavailable) information to be constructed, and their analysis often becomes unfeasible. Instead, qualitative models are much easier to analyze and, although simplified, they can provide useful information about the GRN functioning.

Studying and understanding GRNs is very important. They are the fundamental mechanism at the basis of cell functioning and many diseases are based on perturbation or malfunctioning of some gene regulation activities. Cancer, for instance, is often due to gene mutations that force the unnecessary activation of cell proliferation processes.

Discovering how a gene influences other genes usually requires performing a large number of lab experiments in which cells are placed in different environments or in which their genes are artificially turned on or off to observe how this changes the activation of other genes (i.e. the synthesis of the corresponding RNA and proteins). Once mutual gene influences have been inferred, they can be used to construct a model of the regulatory network that can then be analyzed by using mathematical and computational means. Several databases with gene expression data collected through lab experiments are nowadays available on public databases such as Expression Atlas [73] and Gene Expression Omnibus (GEO) [32].

Many notations and formalisms have been applied to model GRNs. In the first part of the paper, we analyse ODEs and Gillespie's algorithm, which both guarantee a detailed description of the network dynamics. In the second part, we described different logical models that are able to point out the qualitative basic principles that characterize the biological mechanisms under analysis, revealing their emergent behaviour. Among them, the most common approaches are those based on *Boolean networks* [97]. A Boolean network is essentially a set of Boolean variables whose values are periodically updated. The updated value of each variable is a function of its current value and of the values of a number of other variables. Typically, the update process can be either synchronous (all variables are updated at the same time) or asynchronous (one variable at a time is updated). Each Boolean variable represents the activation state of a different gene, and update functions express the influences of other genes. Starting from an initial configuration of active genes, Boolean networks can be used to simulate the evolution of such configuration over time. Moreover, by considering all possible gene configurations, Boolean networks allow key configurations (e.g. attractors) to be identified.

In addition to Boolean networks, several other computer science notations have been successfully applied to the modelling and analysis of biological systems. For example, Hybrid systems [4, 52, 60], process calculi such as the  $\pi$ -calculus [88], the Bio-ambients calculus [87], Bio-PEPA [30], Beta-binders [85], CLS [13], and the  $k$ -calculus [35].

In this paper, as relevant formalisms applied in the context of GRNs, we consider Petri nets [70, 89], P systems [75], and reaction systems [38]. In particular, the qualitative analysis provided by Petri nets can reveal finer detail to metabolic and signalling networks [54], and can therefore be used to examine specific behaviours as the reachability of a given state. Among bio-inspired formalisms, we include P systems, because of their diffusion as modelling tools for biological systems. Finally, we present reaction systems because they represent an innovative and emergent approach [33].

All of these approaches try to capture and describe gene interactions in a different way to provide different viewpoints and enabling different analysis methods. Moreover, several software tools are available for the analysis for GRNs, such as BioTapestry [62], Virtual Cell [61], and GIN-sim [24].

The aim of this paper is to provide a survey on computational approaches to GRN modelling and analysis, by starting from the biological and quantitative modelling background notions, and by describing more in depth qualitative approaches such as Boolean networks and some computer science formalisms.

The paper is structured as follows: in Section 2 we provide the necessary biological background; in Section 3 we recall quantitative modelling approaches, which are based on standard chemical kinetic laws; in Section 4 we describe Boolean networks and their application to the modelling of GRNs; in Section 5 we survey the applications to GRNs of other computer science formalisms such as Petri nets [49, 69, 83], membrane systems [16, 76] and reaction systems [8]; and, finally, in Section 6 we draw our conclusions.

## 2 Biological background

Cells are complex systems made of many connected components having different functions and characteristics [63], as the ability to interact each other by way of a variety of chemical and mechanical signals.

Nearly every cell contains long DNA molecules, essential to build and maintain the organism. A gene is a portion of the DNA, and it is the basic physical and functional unit of a cell. Each cell expresses (i.e. turns on) only a fraction of its genes. This phenomenon, known as *gene regulation*, affects the cell differentiation, representing how a cell becomes of a specialized type. Most genes contain the instructions to synthesize proteins, that are large bio-molecules performing a wide range of activities, such as catalysis of chemical reactions, DNA replication, molecules transportation, cell membrane preservation, among others. Hence, the DNA represents the information storage of the cell, while proteins are the real actuators and contribute actively to the cell functioning.

The information flow from the DNA to proteins is a crucial principle of molecular biology, often called the *central dogma* [31], which consists of two main steps, *transcription* and *translation*, known together as *gene expression*:



Each protein has a corresponding gene in the DNA. Transcription is the biochemical process in which a specific segment of DNA is copied into RNA, by the enzyme RNA polymerase, resulting in a messenger RNA (mRNA) that is a single-stranded copy of one gene. Then, during the translation process, the mRNA is read and used to assemble the chain of amino acids that form a protein.

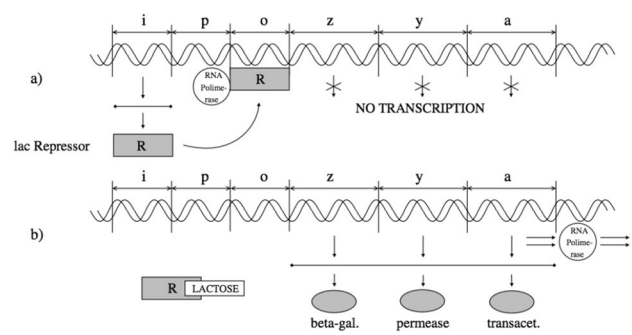
The cell behavior can be modified by processes or external events regulating protein synthesis. Regulation activities can consist in either promoting or inhibiting the DNA transcription or the RNA translation, or by favoring or hampering the chemical reactions in which the proteins are involved. These activities all together constitute the so called *Gene Regulatory Networks (GRNs)*, that hence are the mechanisms that allow cells to react to environmental changes.

## 2.1 Example of GRN: the *lac* operon

*Escherichia coli* is a bacterium usually present in the intestine of many animal species. As most bacteria, it adapts its internal functioning to the environmental changes by modifying the kind of proteins it produces [101, 106]. Among the proteins it synthesizes, there are enzymes necessary for metabolizing nutrients. Different enzymes are involved in the processes (metabolic pathways) executed to extract energy from different nutrients. To save energy, *E. coli* synthesizes the enzymes for lactose metabolism only if lactose is actually present in the environment. The regulation of enzymes for lactose degradation is performed by the group of genes known as the *lac operon*.

The *lac* operon is a sequence of six genes in the DNA of *E. coli* that are responsible for the synthesis of three enzymes involved in the metabolism of lactose. The first three genes of the operon (called *i*, *p*, and *o*) regulate the production of the enzymes, and the last three genes (called *z*, *y*, and *a*, and known as *structural genes*) are transcribed into a single mRNA that is then translated into *beta-galactosidase*, *lactose permease*, the *transacetylase* (the three enzymes). In particular, beta-galactosidase splits lactose into glucose and galactose; lactose permease is a protein that is incorporated in the membrane of the bacterium and actively transports the sugar into the cell; and transacetylase has a marginal role. These three proteins should be synthesized only when lactose is present in the environment.

As shown in Fig. 1, gene *i* is used to synthesize a protein called *lac repressor*. When lactose is not present, the *lac* repressor, binds to gene *o* (called *operator gene*) which is located in the middle of the operon. In this way, the *lac*



**Fig. 1** The regulation process in the *lac* operon. In the presence of lactose (case a) the *lac* repressor binds the gene *o* and precludes the RNA polymerase from transcribing genes *z*, *y*, and *a*. When lactose is present (case b) it binds to and inactivates the *lac* repressor

repressor becomes an obstacle for the RNA polymerase, which is the enzyme responsible for transcribing the DNA into the RNA, and that usually binds the DNA at the location of gene *p* (called promoter gene) to start the transcription.

On the other hand, when lactose is present, it binds to the *lac* repressor and removes it from the DNA, allowing the RNA polymerase to scan the operon and transcribe the genes corresponding to the three proteins for lactose degradation. When lactose has been consumed, the *lac* repressor rebinds to the DNA, and the synthesis of the three enzymes stops.

This process is a small example of GRN, in which the synthesis of proteins is under the control of the *lac* repressor, which acts preventing the expression of other genes.

## 2.2 Inferring GRNs from gene expression data

GRNs control many different aspects of cell development and life, like differentiation, proliferation and metabolism. Moreover, GRNs often involve dozens of genes influencing each other in different ways. For these reasons, understanding the mechanisms underlying these networks is very challenging.

As reviewed in [51, 80], detecting the individual aspects of regulatory interactions requires multiple experimental approaches, combining distinct observations that include:

- Spatial and temporal gene expression data, which necessitate analysis performed at multiple time points and phases of development;
- Identification of functional interactions among genes to determine how these factors regulate each other's expression. In particular, this can be achieved in two ways: by *trans-perturbation*, perturbing the transcription factor, and by *cis-perturbation*, mutating the binding sites for specific genes.
- Identification of physical interaction between transcription factors and binding sites.

The mentioned techniques give us information on different levels of analysis. However, assembling several experimental data does not reveal the functional regulatory interaction, which can only be hypothesized in the generation of a GRN model [2]. Hence, the analysis of GRNs with computational methods, as the ones we will review in the next sections, can also be seen as a way to *validate* GRNs inferred from data against observed behavioral phenomena, as described in [39, 99].

### 3 GRNs as chemical reaction networks

We have seen that GRNs involve processes, such as DNA transcription and translation, binding of proteins with DNA or other molecules, etc., that are essentially chemical reactions. Hence, a GRN can be seen as a chemical reaction network, and standard methods from the theory of chemical reaction kinetics can be used to model the dynamics of such GRN.

A *Chemical Reaction Network (CRN)* is a set of transformations involving one or more chemical species, in a specific situation of volume and temperature [41]. The chemical elements that are transformed are called *reactants*, and those that are the result of the transformation are called *products*. A chemical reaction can be represented as an equation, showing all the species involved in the process. A simple example of chemical reaction is the following:



In this case,  $A$ ,  $B$ ,  $C$  and  $D$  are the species involved in the process:  $A$  and  $B$  are the reactants,  $C$  and  $D$  are the products. The parameters  $a$ ,  $b$ ,  $c$  and  $d$  are called *stoichiometric coefficients* and represent the multiplicities of reactants and products participating in the reaction. The symbol  $k_1$ , referred to as *kinetic constant*, is a positive real number giving information about *how fast* the process occurs.

Actually, according to the *law of mass action*, the rate of a chemical reaction is proportional to its kinetic constant and to the concentrations of its reactants (counted as many times as expressed by their stoichiometric coefficients). For example, the rate of reaction 1 is

$$k_1[A]^a[B]^b \quad (2)$$

where  $[s]$  denotes the concentration of the molecular species  $s$ .

Reaction rates can be used to construct a dynamical model of a CRN in terms of Ordinary Differential Equations (ODEs): one equation for each considered chemical species and having the rates of the reactions as terms. The rate of each reaction will appear as a positive term in the equations of its products, and as a negative term in the equations of

its reactants. Moreover, in each equation the rate will be multiplied by the stoichiometric coefficient of that reactant or product [14].

For example, let us consider again the *lac* operon GRN we introduced in Section 2.1. To describe the model as a CRN, we refer to the simple and illustrative example described in [105], and consisting of the reactions reported in Table 1. Here,  $i$  represents the gene for the inhibitor protein,  $r_1$  the associated mRNA,  $I$  the inhibitor protein, and RNAP is the binding site. For the sake of simplicity, the *lac* operon is represented as a single entity, denoted as  $Op$ , and the mRNA transcript from the operon is denoted by  $r$ , and this codes for all three *lac* proteins.

As explained above, according to the standard mass action kinetics, we obtain the system of ODEs shown in Fig. 2. (See [101, 106, 107] for more details.)

A common method to study systems of ODEs is by means of numerical integration [14, 37], that allows us to obtain the dynamics of the concentrations of all the molecules of the CRN over time, starting from given initial values. In the case of the ODEs of the *lac* operon, by assuming the initial concentrations reported in Table 2, it is interesting to study how the dynamics of the concentration of the beta-galactosidase enzyme (denoted  $Z$  in the ODEs) depends on the availability of lactose. As shown in Fig. 3a, when lactose is present ( $[Lactose]_0 = 1000$ ), beta-galactosidase starts to

**Table 1** The chemical reactions and kinetic constants of the *lac* operon GRN (See [105] for more details)

Chemical reactions	Kinetic constants
$i \xrightarrow{k_1} i + r_1$	$k_1 = 0.02$
$r_1 \xrightarrow{k_2} r_1 + I$	$k_2 = 0.1$
$I + Lactose \xrightleftharpoons[k_4]{k_3} ILactose$	$k_3 = 0.005, k_4 = 0.1$
$I + Op \xrightleftharpoons[k_6]{k_5} IOp$	$k_5 = 1, k_6 = 0.01$
$Op + RNAP \xrightleftharpoons[k_8]{k_7} RNAPo$	$k_7 = 0.1, k_8 = 0.01$
$RNAPo \xrightarrow{k_9} Op + RNAP + r$	$k_9 = 0.03$
$r \xrightarrow{k_{10}} r + Z$	$k_{10} = 0.1$
$Lactose + Z \xrightarrow{k_{11}} Z$	$k_{11} = 0.01$
$r_1 \xrightarrow{k_{12}}$	$k_{12} = 0.01$
$I \xrightarrow{k_{13}}$	$k_{13} = 0.002$
$ILactose \xrightarrow{k_{14}} Lactose$	$k_{14} = 0.002$
$r \xrightarrow{k_{15}}$	$k_{15} = 0.01$
$Z \xrightarrow{k_{16}}$	$k_{16} = 0.001$

$$\begin{cases} \frac{d[i]}{dt} = 0, \\ \frac{d[r_I]}{dt} = k_1[i] - k_{12}[r_I], \\ \frac{d[I]}{dt} = k_2[r_I] - k_3[I][Lactose] + k_4[ILactose] - k_5[I][Op] + k_6[IOp] - k_{13}[I], \\ \frac{d[Lactose]}{dt} = k_4[ILactose] - k_3[I][Lactose] + k_{14}[ILactose] - k_{11}[Lactose][Z], \\ \frac{d[ILactose]}{dt} = k_3[I][Lactose] - k_4[ILactose] - k_{14}[ILactose], \\ \frac{d[Op]}{dt} = k_6[IOp] - k_5[I][Op] - k_7[Op][RNAP] + (k_8 + k_9)[RNAPo], \\ \frac{d[IOp]}{dt} = k_5[I][Op] - k_6[IOp], \\ \frac{d[RNAP]}{dt} = (k_8 + k_9)[RNAPo] - k_7[Op][RNAP], \\ \frac{d[RNAPo]}{dt} = k_7[Op][RNAP] - (k_8 + k_9)[RNAPo], \\ \frac{d[r]}{dt} = +k_9[RNAPo] - k_{15}[r], \\ \frac{d[Z]}{dt} = +k_{10}[r] - k_{11}[Lactose][Z] - k_{16}[Z]. \end{cases}$$

**Fig. 2** ODEs modelling the kinetics of the *lac* operon interpreted as a CRN

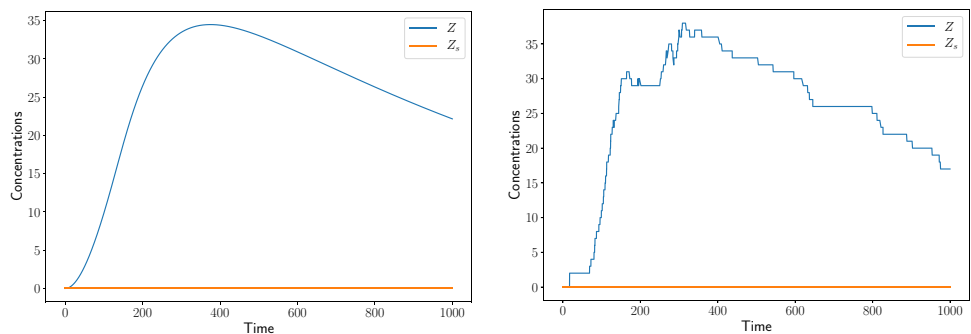
be synthesized, reaching a concentration peak after 350 time units, and then by decreasing back towards zero due to the consumption of all the available lactose. On the other hand, when lactose is absent ( $[Lactose]_0 = 0$ ), the concentration of beta-galactosidase remains low due to the regulation activity performed by the *lac* repressor.

As we have seen, ODEs can be used to study the dynamics of GRNs. However, ODEs suffer from an approximation

**Table 2** Initial concentrations of the *lac* operon CRN ( $[Lactose]_0$  is in the text), given as number of molecules per cell

Initial concentrations
$[i]_0 = 1$
$[r_I]_0 = 0$
$[I]_0 = 50$
$[Op]_0 = 1$
$[IOp]_0 = 0$
$[RNAP]_0 = 100$
$[RNAPo]_0 = 0$
$[r]_0 = 0$
$[Z]_0 = 0$
$[Lactose] = 1000$
$[ILactose] = 0$

**Fig. 3** Simulation results of the *lac* operon model with parameters and initial values as reported in Tables 1 and 2. The graphs show the dynamics of the concentration of beta-galactosidase in the presence (Z) and in the absence (Z<sub>s</sub>) of lactose



**(a)** Numerical integration results

**(b)** Stochastic simulation results

problem that becomes particularly relevant in the case of GRNs. The problem is that ODEs variables are real numbers, assumed to model concentrations of molecules. In GRNs, instead, most of the molecules are genes, which are present in very small numbers (e.g. a single instance) and that change their state (e.g. bound/unbound to some protein) in a discrete way.

In the ODEs modelling the *lac* operon, for example, there are two variables, Op and IOp, corresponding to the concentration of operator when it is unbound and bound to the repressor, respectively. According to the ODEs dynamics, these two variables continuously take values in  $[0, 1] \subseteq \mathbf{R}$ , with  $[Op] + [IOp] = 1$ . A more accurate description of the state of gene Op would instead require one of the two variables to be equal to 1 and the other to 0, with discrete switches between the two opposite configurations.

Although the approximation introduced by ODEs is often considered acceptable to obtain a more accurate quantitative representation of the GRN dynamics it is preferable to switch to the *stochastic simulation* approach [46]. In particular, *Gillespie's Stochastic Simulation Algorithm* [45] (or one of its numerous variants) is the most used method. Gillespie's algorithm is formalized on the basis of the molecular collision theory and it allows simulating the dynamics of chemical reactions like the ones in Fig. 1 (but sometimes with slightly different kinetic constants) by assuming *discrete* quantities of molecules (rather than concentrations) and by considering reaction rates as *stochastic rates* rather than deterministic ones.

In Fig. 3b we show the results of stochastic simulation of the *lac* operon reactions, reported in Fig. 1. As before, in the presence of lactose the quantity of beta-galactosidase enzymes synthesized is higher, but now the dynamics shows irregular peaks due to the stochastic aspects of chemical reactions that are now taken into account.

Observing these examples, we can notice that the quantitative models of GRNs (ODEs and stochastic models) give a precise description of the gene regulation dynamics. Unfortunately, many details and parameters necessary for these models are often not precisely known or can vary



significantly from case to case. Moreover, although optimized and approximate variant of these methods are available [37, 46], these models still have scalability problems because of the problems of numerical integration and stochastic simulation algorithm with stiff systems, and of the huge number of simulations that often have to be performed to investigate the system dynamics.

## 4 GRNs as Boolean models

Quantitative methods, such as the ODEs and stochastic simulation presented in Section 3, are extremely useful to study cell dynamics when mechanistic details and kinetic parameters are known. On the other hand, the noisy nature of biological data makes difficult (often impossible) to exactly determine parameter values [72]. Moreover, analysis of quantitative models is often computationally unfeasible. Then, more abstract (qualitative) models are often preferable, since they can provide useful information on the system dynamics in reasonable times.

### 4.1 Boolean network models

The most common qualitative modelling frameworks for GRNs are based on *Boolean networks* [97], which are able to describe biological phenomena such as oscillations, multi-stationary events, long-range correlations, switch-like behaviour stability and hysteresis. A Boolean network consists of a set of Boolean variables (the nodes of the network) each associated with a Boolean function defined on a subset of the variables. In the context of GRNs, each variable represents the activation state of one gene, which can be 1 (active) or 0 (inactive) [103]. Moreover, the Boolean function associated to a variable represents the influence of other genes on the activation state of the gene described by such a variable. In practice, such a function will be periodically

used to *update* the corresponding Boolean variable, allowing the dynamics of the GRN to be simulated.

As usual, Boolean functions can be equivalently represented as logic formulas with *and*, *or*, and *not* operators, or by truth tables. In this paper we will denote a Boolean function associated to a Boolean variable  $Y$  and defined on Boolean variables  $X_1, \dots, X_n$  as an assignment operation of the form

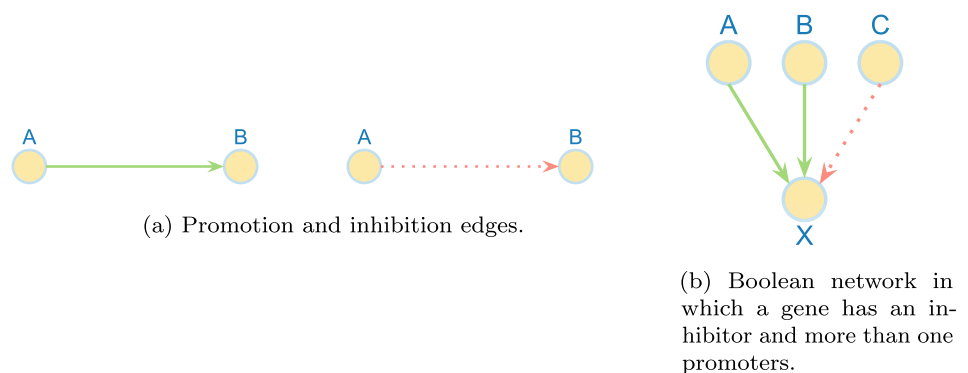
$$Y := f(X_1, \dots, X_n)$$

where  $f$  is expressed as a logic formula.

Boolean networks describing GRNs are often represented as directed graphs, where each node corresponds to a Boolean variable and each edge represents the influence of one variable on another one (that is, the influence of one gene on another gene). Edges can be of two types, *promotion* or *inhibition*, corresponding to a *positive* or *negative* influence, respectively [55]. In the graphical representation of a Boolean network, the two types of edges are usually depicted by using arrows with different shapes or colors. In this paper we will represent promotion edges as green solid arrows and inhibition edges as red dotted arrows, as shown in Fig. 4a.

The graph representation of a Boolean network is actually quite abstract. Let us consider, for example, the graph in Fig. 4b. It represents a network with four genes  $A$ ,  $B$ ,  $C$  and  $X$ , in which  $A$  and  $B$  have a positive influence on  $X$ , and  $C$  has a negative influence on the same gene. This suggests that in the formula expressing the condition for the activation of  $X$  there will be a positive (not negated) occurrence of  $A$  and  $B$  while there will be a negative (negated) occurrence of  $C$ . This is because the truth of  $A$  and  $B$  favors the truth of  $X$ , while the truth of  $C$  disfavors the truth of  $X$ . However, the graph representation is very abstract: it does not specify the way the genes influence each other. Indeed, there are several different Boolean functions, that can correspond to configuration of edges of Fig. 4b, such as

**Fig. 4** Examples of Boolean networks



$X := A \text{ and } B \text{ and not } C$   
 $X := (A \text{ or } B) \text{ and not } C$   
 $X := (A \text{ and not } C) \text{ or } B$   
 $X := \dots$

In all the previous Boolean functions A and B have a positive influence on X, while C has a negative influence on X. As a consequence, several different Boolean networks can correspond to the same graphical representation. On the other hand, starting from a generic Boolean function it is not always possible to obtain a clean and simple graphical representation. While it is easy to do it for formulas in which occurrences of each gene are either all positive (not negated) or negative (negated), for formulas containing a gene having both positive and negative occurrences the translation into graphical representation is not straightforward. This happens, for instance, in the following case:

$$X := (A \text{ and } B) \text{ or } (\text{not } A \text{ and } C)$$

in which A has either a positive or a negative influence on X depending on the activation states of B and C. This formula cannot be trivially represented in graphical form because the type of the arc connecting A with X cannot be determined. A possible solution is to create an extended graph with include nodes representing *combinations* of genes rather than single genes as discussed in [12].

Even if the graph representation does not give any information on the way genes interact, it is often what is obtained from the lab experiments performed to infer a GRN (see Section 2.2). Hence, a critical step for the accuracy of the model is then to choose a consistent method to translate combinations of positive and negative influences into Boolean functions. Indeed, as pointed out in [36], Boolean networks are easy to interpret and they offer a simple dynamic approach for GRN [34]. Moreover, under specific circumstances, the predicted behaviour of Boolean networks is qualitatively similar to that obtained by using an ODE model of the network, as described in [84]. On the other hand, the predictions of Boolean models can become unrealistic for larger networks when compared to those of the corresponding ODE models.

The main limitation of this approach lies with the discretization step, which can cause the loss of important details of the system behaviour. Indeed, gene expression is rarely a matter of full-activation or full-silencing, since there are often different gene states in between. Moreover, discretization forces genes to update their state according to a global clock, while in reality genes change their state with different frequencies as described in [64].

#### 4.1.1 Synchronous vs asynchronous dynamics

The dynamics of a Boolean network model is given as a sequence of steps in which the value of the Boolean variables are updated with the values obtained by the associated Boolean functions. Usually, two alternative updating schemes can be considered: *synchronous* and *asynchronous*.

The *synchronous scheme* is deterministic: all the variables are updated simultaneously according to the associated Boolean functions. As described in [40, 94], this approach, based on discrete time steps, is computationally efficient and easy to implement. Synchronization of updates implicitly assumes that all the biological events have similar duration. This is often quite a strong assumption from the biological viewpoint [43], since in general the timescales of biological events can vary from fractions of seconds to hours [74, 103].

To address this problem we can make use of the *asynchronous scheme*, which assumes that only one variable can be updated in a single step, and every variable is equally likely to be updated [43]. In this approach, the nodes can be updated in any order [27, 50, 103], and the dynamics becomes non deterministic.

The asynchronous update strategy is often considered more realistic, since the underlying chemical reactions have in general different rates, so they happen at different (stochastic) times. In [92], the authors compare the synchronous and the asynchronous schemes to understand which one provides a more realistic description of the biological system under study. They show that the asynchronous update scheme detects the attractors more accurately than the synchronous scheme. On the other hand, the synchronous update strategy is in many case preferred since it is deterministic and makes the analysis of the network behaviours easier.

In [28], the authors propose two new approaches to the analysis of Boolean models, which combine continuous-time techniques with discrete events, to describe the timescales of the genetic processes realistically. By these methods, they show how to overcome the limitation of the synchronous scheme that can be considered potentially unrealistic and may not be suited for intracellular biological processes, due to the variety of timescales.

An approach that is alternative to both the synchronous and the asynchronous strategies is the one offered by *probabilistic Boolean networks* [95, 96]. They are Boolean networks which permit the quantification of the relative influence and sensitivity of genes in their interactions with other genes through the use of probabilities. This allows the network dynamics to be expressed in terms of a Discrete Time Markov Chain, suitable for probabilistic steady state analysis or Monte Carlo simulation. Probabilistic Boolean networks can be still considered as a qualitative modelling framework, although at a slightly lower abstraction level.

On the same line, a remarkable formalism are Bayesian networks, a probabilistic graphical model commonly used for the GRN analysis. This approach is particularly suited for characterizing time-series gene expression data, which are the data results of an experiment in which clusters of genes are measured at different, successive time points. An interesting analysis of this approach is done in [58], where the authors compare the performance of Bayesian networks to Probabilistic Boolean networks, and state that the first one it is able to identify more gene interactions than Probabilistic Boolean networks. For more details about Bayesian networks, we remand to these articles [22, 102].

#### 4.1.2 Attractor analysis

Given an initial configuration for the Boolean variables, the model evolves following either a synchronous or an asynchronous updating scheme. This causes the values of the variables to change describing new configurations (or *states*) reached by the modeled GRN. After a number of steps, the model typically reaches a state (or sequence of states) known as *attractor* [43]. An attractor can be of one of these three types:

- *Self-loop*, which is a single state attractor;
- *Simple loop*, which is a cyclic sequence of states;
- *Complex loop*, which is the alternation of two or more simple loops.

When the synchronous update scheme is adopted, it is only possible to reach self-loops and simple loops, since the dynamics is deterministic. In the asynchronous case, instead, it is also possible to reach complex loops.

The simplest type of attractor, the self-loop, corresponds to a fixed point of the Boolean functions of the networks. All the updates do not change the values of the variables. Since the Boolean functions are computed independently from each other, a self-loop is the same regardless of the chosen updating schemes (synchronous or asynchronous). Moreover, in complex networks, the update schemes influence the probability to reach the same attractors, as shown in [40, 92], where the attractors, reached in the synchronous models, are not present in the corresponding asynchronous models.

Identifying the attractors is biologically very relevant because they are often correlated to the gene activation configurations of specific cellular phenotypes, as shown in [43, 57, 93]. Besides, the analysis of attractors helps to examine the system dynamics and compare it with experimental data.

Detecting the attractors is sometimes challenging, in particular, when the exact initial configurations are unknown or when large Boolean models are studied. As regards the first aspect, the problem is that it requires testing the model by

considering  $2^n$  possible initial configurations, with  $n$  is the number of variables in the network. An approach to overcome this problem is to sample a large number of initial configurations and to calculate the probability to reach certain attractors, as in [59, 109].

For larger models, it can be computationally unfeasible to identify the attractors [110], in particular when the asynchronous updating scheme is adopted. An interesting approach to this problem is described in [43], where the authors propose a combined synchronous-asynchronous traversal technique to find the attractors of an asynchronous model in run time proportional to the synchronous scheme. Several other works propose methods for reducing the network to simplify the model analysis. In [17, 71, 92], the authors remove certain variables (called *frozen nodes*) that evolve to the same steady state independently of their initial values and thus are not relevant for attractor identification [103].

#### 4.1.3 Examples of GRNs as Boolean networks

As shown in [3], a Boolean network can accurately predict the dynamics of a biological system. For this reason, Boolean networks have been applied to model several biological regulatory networks, such as the yeast cell cycle [57], the differentiation of T-helper [68], the signal transduction network for abscisic acid-induced stomatal closure [59], the mammalian cell cycle [40], and the expression of *Drosophila* segment polarity genes [3]. In this paper we show how to use GRNs to model the *lac* operon example we presented in Section 2.1.

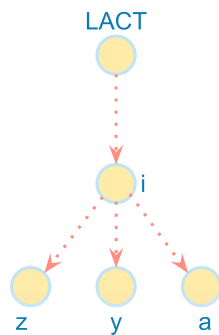
The *lac* operon example. As often happens in practice to model the *lac* operon regulation network described in Section 2.1, we start by introducing the graph representation of the Boolean network (see Section 4.1). We associate each gene (and also lactose, which is the initial stimulus) to a node, which is connected to the others nodes by edges describing gene interactions. The result is the graph in Fig. 5, which shows that the three genes responsible for lactose metabolism,  $z$ ,  $y$  and  $a$ , are negatively influenced by gene  $i$ , from which the *Lac* repressor is synthesized. Moreover, gene  $i$  is in turn negatively influenced by lactose, which binds and inhibits the *Lac* repressor.

We remark that the negative influence of lactose on the gene  $i$  is, in reality, very different from the negative influence of the gene  $i$  on  $z$ ,  $y$  and  $a$ . The former is at the protein level (interference with the activity of the *lac* repressor) while the latter is at the DNA level (interference with gene transcription). However, this difference is not captured at the abstraction level of a qualitative model.

To each Boolean variables, we associate a Boolean function, described as the assignment of a Boolean expression. The expression has to be consistent with the influences



**Fig. 5** The Boolean model of the *lac* operon. The lactose has a negative influence on the gene *i*, which has a negative influence on the genes *z*, *y*, and *a*



expressed by the edges of the graph representation. However, in this case each gene is influenced by only one other gene, so defining the Boolean functions is straightforward:

```
Lact := Lact
i := not Lact
z := not i
y := not i
a := not i
```

Let us now consider an initial configuration in which lactose is present, that corresponds to the following initial values for the Boolean variables:

$$\text{Lact} = 1 \quad i = 1 \quad z = 0 \quad y = 0 \quad a = 0$$

By adopting a synchronous updating scheme, all variables are updated at each step. Then, by simulating the network, we obtain the sequence of steps described in the following table.

	Lact	<i>i</i>	<i>z</i>	<i>y</i>	<i>a</i>
Step 0	1	1	0	0	0
Step 1	1	0	0	0	0
Step 2	1	0	1	1	1

As we can notice, the configuration reaches a stable state (self-loop attractor) at step 2, in which gene *i* is not active (i.e. the *Lac* repressor it synthesizes is blocked by the binding with lactose) and genes *z*, *y* and *a* are active (i.e. the enzymes for lactose metabolism are synthesized). This corresponds to what already observed with the quantitative models described in Section 3.

Keeping the same initial configuration, we can adopt also the asynchronous updating scheme, in which a single variable is updated at each time step. Now the dynamics is no longer deterministic. A possible evolution is given by the sequence of steps described in the following table in which,

as we can notice, the configuration reaches a stable state (self-loop attractor) at step 4.

	Lact	<i>i</i>	<i>z</i>	<i>y</i>	<i>a</i>
Step 0	1	1	0	0	0
Step 1	1	0	0	0	0
Step 2	1	0	0	1	0
Step 3	1	0	1	1	0
Step 4	1	0	1	1	1

In this case, both the updating schemes reach the same stable state, but, as described in Section 4.1.1, this is not always the case since in general the two approaches can lead to very different configurations.

### 4.2 Threshold Boolean networks

A particular class of Boolean networks are the *threshold Boolean networks* [19, 108]. Their main characteristics is in the way they infer Boolean update functions from the graphical representation of the network. The idea is that each edge is weighted by an integer number and each node is associated to a threshold value. The Boolean function for the update of a variable corresponding to a given node is obtained by

computing a weighted sum of the values of the influencing nodes. The sum is then compared with the threshold to determine the new value of the Boolean variable.

This way of specifying the Boolean functions of the network is a simple and deterministic, and solves the problem of mapping graph representations into actual Boolean networks we mentioned at the end of Section 4.1. On the other hand, this is not a general methodology since not all Boolean functions can be specified in terms of weighted sums.

To define threshold Boolean networks, let us denote the set of nodes as  $M = \{S_1, S_2, \dots, S_n\}$ . Each node is also a Boolean variable and can take 0 and 1 as values. As before, these values represent the inactive and the active states of the gene modeled by the node. Edges represent positive or negative influences between genes, corresponding to positive or negative weights. In this paper, we assume the possible weights to be restricted to 1 and  $-1$  (although this restricts the expressive power), and we denote with  $a_{ij}$  the weight of the edge connecting node  $S_i$  to node  $S_j$ . Let  $E$  be the set of all weighted edges  $a_{ij}$ , the pair  $(M, E)$  is called *threshold Boolean network* [8].

Another peculiarity of threshold Boolean networks is that nodes in  $M$  are partitioned into

*self-activating* ( $M_{sa}$ ) which remain active until not inhibited by some other node;

*non self-activating* ( $M_{nsa}$ ) which become inactive if not sustained by some other node. The state of node  $i$  at time  $t + 1$ , denoted  $S_i(t + 1)$ , is computed from the states of all nodes at time  $t$  as follows:

$$S_i(t + 1) = \begin{cases} 1 & \text{if } \sum_j a_{ij}S_j(t) > \theta_i \\ 0 & \text{if } \sum_j a_{ij}S_j(t) < \theta_i \\ S_i(t) & \text{if } S_i \in M_{sa} \wedge \sum_j a_{ij}S_j(t) = \theta_i \\ 0 & \text{if } S_i \in M_{nsa} \wedge \sum_j a_{ij}S_j(t) = \theta_i \end{cases}$$

where

- $a_{ij} > 0$  if node  $j$  promotes node  $i$  (we assume  $a_{ij} = 1$ );
- $a_{ij} < 0$  if node  $j$  inhibits node  $i$  (we assume  $a_{ij} = -1$ );
- $a_{ij} = 0$  if node  $j$  has no influence on node  $i$ ;
- the value  $\theta_i$  is the *threshold parameter* for node  $i$  (often equal to 0).

As a first example of threshold Boolean network consider the graph depicted in Fig. 5. Assume that all nodes are self-activating; formally we have  $M = \{LACT, i, z, y, a\}$ ,  $M_{sa} = \{LACT\}$  and  $M_{nsa} = \{i, z, y, a\}$ , and assume weights ( $a_{ij}$ ) equal to  $-1$  for all the edges. By setting the threshold parameter for  $i, z, y$  and  $a$  to  $-1$ , and 0 for  $LACT$ , we obtain a threshold Boolean Network model for the *lac* operon.

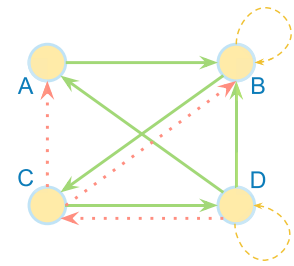
Consider again the case where  $LACT$  and  $i$  are both initially active. After the first step (Step 1 in the table below), gene  $i$  becomes inactive because  $-1 \not> -1$ ,  $LACT$  is active because it is self activating and  $z, y, a$  are inactive because  $i$  was active at the previous step and  $-1 \not> -1$ . At Step 2,  $z, y$  and  $a$  become active because  $i$  was inactive at the previous step and  $0 > -1$ . The following table reports the evolution of the Boolean network

	Lact	$i$	$z$	$y$	$a$
Step 0	1	1	0	0	0
Step 1	1	0	0	0	0
Step 2	1	0	1	1	1

An additional example of threshold Boolean network in graph representation is given in Fig. 6. In addition to standard Boolean networks, we have also non self-activating nodes that are decorated by a (yellow) dashed arrow self loop.

In the network in Fig. 6, nodes are  $M = \{A, B, C, D\}$  (with  $M_{sa} = \{A, C\}$  and  $M_{nsa} = \{B, D\}$ ) and the edges are as depicted in the figure. Thus,  $A$  and  $C$  are self-activating, while  $B$  and  $D$  are not. Assuming that the threshold parameter

**Fig. 6** An example of threshold Boolean network representing a GRN with four genes. Weights of edges are absent since we assume weight 1 for positive influences and  $-1$  for negative ones



for each node is 0, we describe the temporal evolution of the network by considering a *synchronous* update scheme and an initial state in which only  $D$  is active. We obtain the sequence of steps shown in the following table.

	A	B	C	D
Step 0	0	0	0	1
Step 1	1	1	0	0
Step 2	1	1	1	0
Step 3	0	0	1	1
Step 4	0	0	0	1

At the first step,  $D$  stimulates the activation of  $A$  and  $B$  because  $C$  (which is their inhibitor) is not present. Since  $D$  is non self-activating, at the second step it is inactive. Then, at the third step  $C$  is activated by  $B$ , which in turn remains active thanks to  $A$ , which in turn remains active because it is self-activating and at step 2  $C$  was inactive. At the fourth step,  $C$  activates  $D$  and inhibits  $A$  and  $B$ , which become inactive. Finally,  $C$  becomes inactive because inhibited by  $D$ . The last state coincides with the first one, so we reached the end of a cycle (simple loop attractor). Different dynamics can be obtained by starting from different configurations [8].

It is worth noting that the Boolean function expressing the activation of a node  $i$  in a threshold Boolean network can easily be defined. Consider a combination of active/inactive genes such that  $S_i(t + 1) = 1$  holds. The conjunction of all active genes in the combination together with the conjunction of the negation of each gene that was inactive gives a formula to obtain the activation of node  $i$ . By taking the disjunction of such formulas for all possible combinations of active/inactive genes such that  $S_i(t + 1) = 1$ , we obtain the Boolean function for the activation of  $i$ . On Section 5.3 such ideas are used to propose a naive translation of threshold Boolean network into Reaction Systems.

### 4.2.1 GRNs as threshold Boolean networks

To show an example of real GRN modeled as a threshold Boolean network, we recall here the model of the yeast (*S. cerevisiae*) cell cycle proposed by Li et al. in [57].

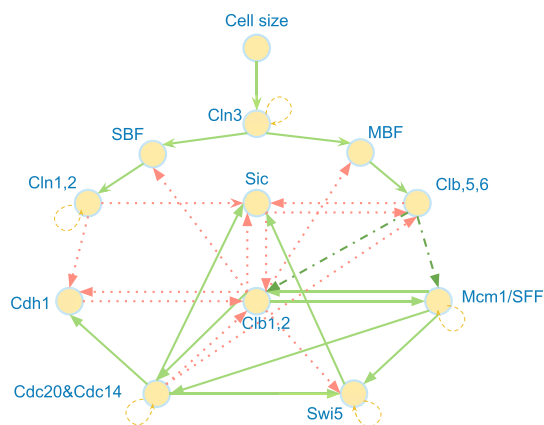
The yeast cell cycle control circuit is one of the best understood molecular control networks, studied as example of robust dynamical process in the cell [19, 29, 66]. Indeed, this process, by which one cell grows and divides into two daughter cells, is a vital biological process the regulation of which is highly conserved among the eukaryotes. It consists of 4 phases:

- $G_1$  (in which the cell grows and, under appropriate conditions, commits to division);
- $S$  (in which the DNA is synthesized and chromosomes replicated);
- $G_2$  (a “gap” between S and M);
- $M$  (in which chromosomes are separated and the cell is divided into two).

After the  $M$  phase, the cell enters the  $G_1$  phase, hence completing a “cycle”.

Even if this network involves more than 800 genes, Li et al. in [57] propose a network of 11 key genes (plus a signal node) shown in Fig. 7. Although in principle the biological events happening during the cell cycle have very different time scales of action, in this system there are gene activation configurations that are known to be milestones for the activation of the different phases of the cycle. This motivates the adoption of the *synchronous update scheme* for the analysis of the system dynamics.

In [57], the model has been used to conduct an exhaustive attractor analysis. The authors considered all the  $2^{11}$  initial states they lead to seven stationary states (self loop attractors). Moreover, one of these seven states attracts 86% of the



**Fig. 7** Boolean network model for the yeast cell cycle control network as defined in [57]

initial states. The genes that turn out to be active in such an attractor are those which have been observed to be expressed in phase  $G_1$  of the yeast cell cycle (that is also the resting phase of the cycle).

The work proposed by Li et al. has the merit of show how modelling real biological genetic circuits can predict sequence patterns of protein and gene activity, as observed in living cells, with much less input (e.g. parameters) than other approaches, such as ODEs. In addition, another advantage of this approach is the possibility to observe that the cell cycle of the regulatory network in yeast is extremely stable and robust, analysing the topology of the network and its response to small perturbations.

## 5 Computational models of GRNs

In literature there are many formalisms that can be used to describe a biological systems at different abstraction levels. In this section, we describe how GRNs can be modeled by using Petri nets, P systems, and reaction systems.

### 5.1 Petri nets

They are a rigorous mathematical formalism providing also an intuitive graphical representation, proposed by Carl Adam Petri [82]. They were originally used for describing, designing and studying discrete event-driven dynamical systems that are characterized as being concurrent, asynchronous, distributed, parallel, random and/or non-deterministic, but they have been extended in many different directions.

Petri nets have been successfully used to model biological systems (see [49, 69, 83] for an introduction) and in particular metabolic networks. Petri nets allow modelling quantitative dynamical aspects like mass flow in a network.

In general, Petri nets can be represented as directed graphs consisting of two different kinds of node: *places* and *transitions*. The graph is bipartite, hence each edge connects a place to a transition, or vice versa. The game of *tokens* models the dynamical aspects of the model. Each place can contain a number of tokens and transitions take tokens from places and give token to other places according to edges orientation. Each edge has a *weight* that specifies how many tokens are taken/given by the corresponding transition. If each source place contains a sufficient number of tokens, then the transition is *enabled* (i.e. it can be *fired*) and the tokens travel along the edges. The state of the system is represented by the allocation of tokens to the places and it is called *marking*. A Petri net comes equipped with an *initial marking* which is the initial allocation of tokens to places.

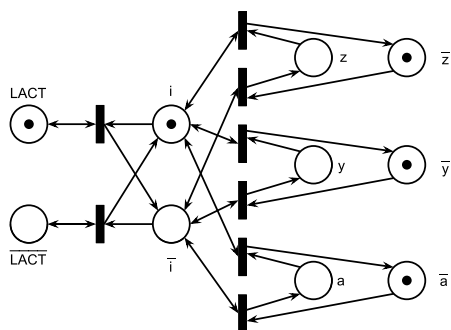
An important advantage of using Petri nets for modelling biological networks is that they are supported by theoretically well-founded techniques and tools for simulation and

analysis. Indeed, a Petri net can be automatically checked for the following properties (see [26] for more details):

- *Boundedness* ensures that the number of tokens in the initial marking and in the evolution of the net for each place is bounded. For metabolic networks this means that products cannot accumulate;
- *P-invariants* are sets of places for which the weighted sum of tokens is constant independently of the sequence of firings. In metabolic networks, this property corresponds to a mass conservation law;
- *T-invariants* are firing sequences which reproduce a marking. In biological terms, T-invariants may represent cyclic behaviours;
- *Reachability* of a marking  $M$  asserts that there exists an evolution from the initial marking to a given marking  $M$ . This property may be relevant for biological networks, as it ensures the existence of an evolution leading the system from an initial state to a desired state;
- *Liveness* ensures that it is always possible to ultimately fire any transition. Liveness guarantees that a reaction can eventually occur.

Petri nets can be used to model GRNs in a *qualitative* way. Due to their *asynchronous* nature, the adoption of an approach analogous to the asynchronous update scheme is rather straightforward. The only non-trivial aspect is that the model has to include one place for each *gene state* (hence, two per gene).

In Fig. 8, we show a Petri net modelling the *lac* operon GRN, corresponding to the Boolean network represented in Fig. 5. Intuitively, the implementation is done as follow: each node of the Boolean network corresponds to two complementary distinct places  $p_i$  and  $\bar{p}_i$ , representing its active and inactive states, respectively. As a consequence, we have the constraint that the sum of tokens in each pair of places  $p_i$  and  $\bar{p}_i$  has to be 1. To draw the transitions to  $p_i$ , we have to consider all the combinations of active/inactive genes that activate  $p_i$  in the Boolean network. For each combinations



**Fig. 8** The initial marking of the Boolean Petri net of the *lac* operon. For each biological entity, we draw an active and inactive places

we draw the transition to  $p_i$ . Moreover, we have to draw all the transitions from all the combinations that do not activate  $p_i$  to  $\bar{p}_i$ . In Fig. 8, we draw the Petri net of the *lac* operon. In this case, the initial marking describes the configuration in which lactose is present, gene  $i$  is active while genes  $z$ ,  $y$  and  $a$  are not. According to the enabled transitions, the token in place  $i$  will be moved into place  $\bar{i}$ , and then the tokens in  $\bar{z}$ ,  $\bar{y}$  and  $\bar{a}$  will be moved into  $z$ ,  $y$  and  $a$ , respectively, describing the activation of the genes for lactose metabolism.

The use of one place for each gene state is necessary to model inhibitory influences between genes. Petri net transitions, whose firing models gene state changes, are triggered by the *presence* of some tokens in the proper places. If we used one place for each gene, with tokens representing gene activation, then a transition modeling a state change (negatively) influenced by an inhibitor would have to be triggered when no token is present in the place representing such an inhibitor. However, standard Petri nets cannot be used this way since they are not able to test for the *absence* of tokens in a place. This possibility is instead offered by Petri nets with inhibitory arcs [23] that are, however, in general much more difficult to analyze because of their increased expressive power.

In [98], the authors proposed a new technique for constructing qualitative and *synchronous* Petri net models of GRNs. Following the approach originally proposed in [25, 90], the Petri net is constructed by directly translating the Boolean formulas into appropriate Petri net control structures to obtain a compact net that correctly captures the original Boolean behaviour of a GRN. Such control structures contain a number of places and transitions specifically added to force the synchronous update strategy. The authors apply their method to the GRN controlling sporulation in the bacterium *Bacillus subtilis* and use simulation and model checking tools to verify hypotheses on the behaviour of the system. The main advantage of the approach proposed in [98] is the possibility to handle incomplete and/or inconsistent behavioural information exploiting the possibility of Petri nets of specifying alternative (non-deterministic) behaviours, even in the case of synchronous GRNs. Moreover, Petri nets modelling enables the application of several analysis methodologies and tools that are available for this formalism.

*Quantitative* models of biochemical and metabolic networks, can be constructed by using *continuous Petri nets* [44]. They are an extension of Petri nets in which the marking of a place is no longer an integer, but a positive real number (called token value) representing the concentration of chemical species. Moreover, each transition is associated to a kinetic constant. These features make continuous Petri nets fully capable of representing chemical reactions. Furthermore, the dynamics of a continuous Petri net can be expressed in terms of ODEs in agreement with the standard

mass action kinetics of chemical reactions that we have described in Section 3.

For the quantitative modelling of gene regulatory networks, Goss and Peccoudv proposed an approach to model stochastic systems in molecular biology using *stochastic Petri nets* [47, 48]. Their approach was illustrated with examples of models of genetic and biochemical phenomena by using existing software. In their works they use stochastic Petri nets also to analyze the stabilizing effect of the protein Rom on the genetic network controlling COLE1 plasmid replication [48]. This approach is useful because it shows how to model biological processes that cannot be modelled suitably by a deterministic approach.

Representing continuous values such as the concentration of mRNA or proteins is an essential factor in a quantitative modelling approach. We have seen that this is captured for instance by continuous Petri nets, but not by ordinary Petri nets. For this reason, in [1, 65, 67] *hybrid Petri nets* are proposed, which extend Petri nets to allow handling continuous factors without explicitly including them in the model. In hybrid Petri nets it is possible to express the relationship between continuous and discrete values, while keeping the characteristics of ordinary Petri nets soundly.

### 5.2 P systems

Although they have been proposed as a model of computation, P systems are also well-established as a modelling notation for biochemical pathways and GRNs [16, 76], in particular in the context of *systems biology* [42, 78]. Indeed, since P systems rules are inspired by chemical reactions, their application to the modelling of chemical reactions is quite straightforward. Moreover, the membrane structure of P systems enable the development of compartmental models. To this aim, several quantitative extensions of P systems have been developed, in both the deterministic [77] and the stochastic [78, 79] settings to carefully take reaction rates into account. Formally, a P System is a tuple:

$$\Pi = \langle \Sigma, \mu, w_1 \dots w_n, R_i \rangle$$

where

- $n$  is the number of membranes;
- $\Sigma = \{c_1, \dots, c_m\}$  is the alphabet of the biological objects (such as proteins, RNA and DNA, chemical species that are involved in the system);
- $\mu$  is the membrane structure, composed of  $n \geq 1$  other membranes;
- $w_1 \dots w_n$  are strings over  $\Sigma$  representing the multisets of objects present in the regions of the membrane structure;
- $R_i$  is a finite set of multiset rewriting rules associated to each membrane.

Thus, the main elements of this formal language are: membranes (that create compartments used to distribute computations); multisets (abstractions of chemical entities that are used as data); evolution (rewriting) rules (abstractions of chemical processes that are used as programs).

The dynamics of a P systems starts from the initial configuration represented by multisets  $w_1, \dots, w_n$ , and evolves through steps in which rules are applied in a *maximally parallel way*. This means that at each step several rules can be applied at the same time (an the same rule can be applied more than once) to different objects in a maximal way. This form of parallel rule application has two implications: (i) rules compete for objects, and (ii) state changes represented by different rules are synchronous.

In the context of GRN, by assuming objects to represent gene activation states, we have that (i) and (ii) together make the modelling of gene state changes difficult. On the one hand, we are forced to adopt a synchronous update strategy, and on the other hand a single object representing the activation state of one gene cannot be used to promote the activation of more than one other genes. For this reason, the modelling of GRN is made more natural by the use of P systems with promoters and inhibitors [20]. This extension allows expressing positive and negative influences between genes as rule promoters and inhibitors, respectively, which are object that condition the application of the rule without being consumed.

Using this extension, the lac operon GRN can be modelled by following P system:

$$\Pi = \langle \{LACT, i, z, y, a\}, [\_ ]_1, \{LACT, i\}, R \rangle$$

where  $R$  consists of the following rules:

$$\epsilon \rightarrow i \mid \overline{i, LACT} \quad i \rightarrow \epsilon \mid LACT \quad \epsilon \rightarrow z \mid \overline{z, i} \quad z \rightarrow \epsilon \mid i$$

where  $\epsilon$  represents the empty multiset and in which rules analogous to the last two, with  $z$  replaced by  $y$  and  $a$ , are omitted.

Each pair of rules represents the activation and inactivation of a gene. Promoters and inhibitors (listed on the right side of  $|$ , with inhibitors denoted with an over-line) express the conditions for the application of the rule. Hence, gene  $i$  is activated when both  $i$  and  $LACT$  are not present (first rule), and it is inactivated when  $LACT$  is present (second rule). Similarly,  $z$  is activated when both  $z$  and  $i$  are not present, and  $z$  is inactivated when  $i$  is present.

In [91], the *lac* operon is presented as a case study of GRN modelling with P systems, further extended in order to take quantitative modelling aspects in to account. For each implemented membrane four different initial conditions are considered: with/without glucose and with/without lactose. Moreover, the authors associate to each rule a finite set of attributes which are meant to capture the quantitative



aspects that are often necessary to characterise the reality of the phenomenon to be modelled like kinetic or stochastic constants. By using this approach, the authors are able to describe the behaviour of the system, and, in addition to the other *lac* operon models, they take into account also the key role played by membranes in the structure and functioning of the cells and the discrete and concurrent character of processes in bio-systems.

Moreover, in [16], the authors proposed a quantitative model of the GRN underlying the *quorum sensing* ability of *Vibrio fischeri*. Quorum sensing is the ability of some bacteria to perceive their own density in the environment in order to express some proteins in a coordinated way, only when their population is large enough. By the proposed approach, the authors can track the behaviour of each individual bacterium in the colony. On the same line, in [15] Bernardini et al. proposed a new variant of P systems inspired by quorum sensing and investigate its computing power.

Some papers proposed approaches that make a synergistic use of quantitative and qualitative methods. In [53], Hinze et al. proposed a bridge between quantitative and qualitative modelling of GRNs by describing a transformation of Hill kinetics for GRNs into P systems. Moreover, in [5, 86], the authors proposed a quantitative model of allelic gene network regulations based on P transducers, that are an extension of P systems. The model copes discrete aspects of gene regulation, such as the one-to-one interactions with DNA molecules, with some continuous aspects, such as those related to the relationship between gene structure and functional organization.

From the purely qualitative viewpoint, an extension of P systems, called *logic network dynamic P systems (LNDP systems)* has been proposed in [100] for the modelling of GRNs in which relationships between genes are inferred by applying the LAPP logic method [104].

Finally, in the context of synthetic biology, Konur et al. in [56] illustrate the use of the Infobiotics Workbench [18], a modelling tool based on stochastic P systems, on some example of synthetic genetic Boolean gates. The authors based their tool on P systems, because this formalism allows the specification of sets of reactions in multiple compartments and transport of molecules among them. Thus, this approach can facilitate the modelling and rapid prototyping of multi-compartment systems.

### 5.3 Reaction systems

Reaction systems [21, 38] were introduced by Ehrenfeucht and Rozenberg as a novel model for the description of biochemical processes driven by chemical reactions occurring inside living cells. Reaction systems are based on two opposite mechanisms: *facilitation* and *inhibition*. Facilitation means that a reaction can occur only if all of its reactants are present, while

inhibition means that the reaction cannot occur if any of its inhibitors is present. The state of a reaction system consists of a finite set of objects which can evolve by means of application of reactions. The presence of an object in a state expresses the fact that the corresponding biological entity, in the real system being modeled, is present. Quantities (or concentrations) of the entities are not described: reaction systems are hence a *qualitative* modelling formalism.

As described in [33], the approach of reaction systems is useful to analyze how the state of the system changes over discrete time steps. Indeed, the evolution of the reaction system is regulated by a deterministic assumption, since all the reactions take place without any concurrency on the consumption of elements, and the next step of the system is calculated considering its current state.

We recall the basic definition of reaction system [21, 38]. Let  $S$  be a finite set of symbols, called objects. A *reaction* is formally a triple  $(R, I, P)$  with  $R, I, P \subseteq S$ , composed of *reactants*  $R$ , *inhibitors*  $I$ , and *products*  $P$ . Reactants and inhibitors are assumed to be disjoint ( $R \cap I = \emptyset$ ), otherwise the reaction would never be applicable. The set of all possible reactions over a set  $S$  is denoted by  $\mathcal{R}(S)$ . A *reaction system* is a pair  $\mathcal{A} = (S, A)$ , where  $S$  is a finite support set, and  $A \subseteq \mathcal{R}(S)$  is a set of reactions. The *state* of a reaction system is described by a set of objects.

Starting from an initial state, the evolution of a reaction system is based on application of reactions. A reaction can be applied if, in the current state, reactants  $R$  are present and inhibitors  $I$  are absent. The result is that products  $P$  will be present in the next state. The application of multiple reactions at the same time occurs without any competition for the reactants (*threshold supply assumption*). Therefore, each reaction which is not inhibited can be applied, and the result of the application of multiple reactions is cumulative.

In [7], the authors define a reaction system modelling the *lac* operon. Considering again Fig. 1, they are interested in the production of the three enzymes for lactose degradation, denoted as a single entity  $Z$ , on which the repressor  $R$  has a negative influence. Then, the repressor  $R$  is in turn subject to the negative influence of lactose (denoted  $LACT$ ). In addition to these influences, they model the activity of the RNA polymerase (denoted  $POL$ ), which can bind to the promoter gene  $p$  and the result of such binding is denoted as  $Pp$ . As a result, they obtain the following simple reaction system:

$$\mathcal{A}_{Operon} = (\{LACT, p, POL, Pp, R, Z\}, \{a_1, \dots, a_6\})$$

$$\begin{aligned}
 a_1 &= (\{p\}, \emptyset, \{p\}) \\
 a_2 &= (\{POL\}, \emptyset, \{POL\}) \\
 a_3 &= (\{LACT\}, \emptyset, \{LACT\}) \\
 a_4 &= (\{POL, p\}, \emptyset, \{Pp\}) \\
 a_5 &= (\{Pp\}, \{R\}, \{Z\}) \\
 a_6 &= (\emptyset, \{LACT\}, \{R\})
 \end{aligned}$$

Rules  $a_1$ ,  $a_2$ , and  $a_3$  describe the persistence of the promoter gene, of the RNA polymerase, and of lactose. Rule  $a_4$  and  $a_5$  describe the activity of the RNA polymerase, namely the binding with the promoter gene and the transcription of the structural genes. The latter rule has  $R$  as inhibitor to model the negative influence of the repressor on the transcription and synthesis of the the three enzymes denoted  $Z$ . Finally, rule  $a_6$  models the negative influence of lactose on the repressor.

Recently, a systematic translation of threshold Boolean networks into *closed reaction systems* has been proposed [8, 9]. The dynamics of the reaction system obtained from the translation simulates the evolution of the Boolean network. This correspondence has two main advantages: on one hand it allows to “play” with the reactions (for example by silencing some genes) to deeply understand the behaviour of the network; on the other hand, it enables the application of techniques to detect causality relationships between genes [10, 11]. Moreover, ancestors formulas, as defined in [6], allow characterizing all the initial configurations leading to a specific attractor.

To translate a threshold Boolean network into a reaction system the following convention is adopted: a gene appears in a state if and only if that gene is active. This allows us to describe configurations of active/inactive genes by means of the corresponding set of active genes.

A naive translation of threshold Boolean networks into reaction systems could be obtained by simply exploiting the truth table expressing the activation function of each gene. The idea is to consider all the combinations of active/inactive genes that lead to the activation of a certain gene in the threshold Boolean network. Consider, for example, the threshold Boolean Network depicted in Fig. 9, with  $E$  non-self activating, weight 1 for positive influences and  $-1$  for

negative ones and threshold parameter equal to 0. The same figure shows also the corresponding Boolean function for the update of  $E$  considering all the different configurations of activation/deactivation of genes  $A, B, C$  and  $D$ .

Consider for example the fifth row of the table on the left which describes a configuration in which just gene  $B$  is active while all the others are inactive. In the case described by this configuration, according to the behaviour of the threshold Boolean network on the left,  $E$  becomes active because  $B$  that has a positive influence is active and all inhibitors are inactive, therefore  $1 > 0$ . This result is reported in the truth table by setting gene  $E$  equal to 1 in the row corresponding to the described configuration.

From such table, a reaction system can be easily obtained by considering all rows that lead to the activation of  $E$ . In particular, the configuration  $A = 0, B = 1, C = 0, D = 0$  described by the fifth row can be translated into the reaction  $(\{B\}, \{A, C, D\}, \{E\})$  which says that  $E$  will be added to the next state (the gene will become active) if  $B$  was present in the previous state and  $A, C$  and  $D$  were not present. From this naive translation we obtain the following five reactions, one for each row leading to the activation of  $E$ :

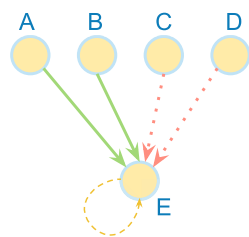
- $(\{B\}, \{A, C, D\}, \{E\})$
- $(\{A\}, \{B, C, D\}, \{E\})$
- $(\{A, B\}, \{C, D\}, \{E\})$
- $(\{A, B, D\}, \{C\}, \{E\})$
- $(\{A, B, C\}, \{D\}, \{E\})$

It is easy to see that this set of reactions can be simplified. For example, from reaction  $(\{A\}, \{B, C, D\}, \{E\})$  and  $(\{A, B\}, \{C, D\}, \{E\})$  we can conclude that  $E$  is produced if  $A$  was present and  $C$  and  $D$  were absent *independently* from the presence of  $B$ . By iterating this process we obtain this set of reactions:

- $(\{A\}, \{C, D\}, \{E\})$
- $(\{B\}, \{C, D\}, \{E\})$
- $(\{A, B\}, \{C\}, \{E\})$
- $(\{A, B\}, \{D\}, \{E\})$

In [9], it is shown that this simplified set of reactions can be obtained directly from the graphical representation of

**Fig. 9** Example of threshold Boolean network (left side) and truth table of the corresponding Boolean function for the update of  $E$  (right side)



A	B	C	D	E
0	0	0	0	0
1	0	0	0	1
0	0	0	1	0
1	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0

A	B	C	D	E
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	0	1
1	1	1	0	1
1	1	1	1	0

the threshold Boolean network, without passing through the construction of the truth tables of the Boolean update functions. The translation of threshold Boolean networks proposed in [9] is guaranteed to produce a reaction system correctly mimicking the behaviour of the network and containing reactions with a minimal number of objects.

The translation of a threshold Boolean network  $(M, E)$  with nodes  $M = \{S_1, S_2, \dots, S_n\}$  and thresholds  $\Theta = \{\theta_1, \theta_2, \dots, \theta_n\}$  is defined as follows. For each node  $S_i$  of the threshold Boolean network, let  $Act(S_i)$  denote the set of nodes that activate  $S_i$ , and  $In(S_i)$  the set of nodes that inhibit it.

The goal of the translation is to produce a set of reactions (a reaction system) able to precisely mimic the behaviour of the Boolean network. The result of the translation of  $(M, E)$  is the closed reaction system  $\mathcal{RS}((M, E)) = (M, A)$ , with a set of reactions  $A$  obtained by applying the following inference rules, where  $\#_x$  denotes the set size.

$$\begin{array}{l}
 P_i \subseteq Act(S_i) \quad I_i \subseteq In(S_i) \quad S_i \in M_{sa} \quad P_i \subseteq Act(S_i) \quad I_i \subseteq In(S_i) \\
 1) \frac{\#P_i - \#(In(S_i) \setminus I_i) = \theta_i + 1}{(P_i, I_i, \{S_i\}) \in A} \quad 2) \frac{\#P_i - \#(In(S_i) \setminus I_i) = \theta_i}{(P_i \cup \{S_i\}, I_i, \{S_i\}) \in A}
 \end{array}$$

The threshold Boolean network  $(M, E)$  is simulated by the execution of a closed reaction system  $\mathcal{RS}((M, E))$  whose reactions are obtained by applying either the inference Rule 1 or the inference Rule 2.

In a Boolean network  $(M, E)$  the node  $S_i$  (at time  $t + 1$ ) is activated whenever at time  $t$  the number of active nodes  $P_i$  activating  $S_i$  minus the number of the active nodes  $H_i$  inhibiting  $S_i$  is greater than  $\theta_i$ . Intuitively, each (minimal) possible combination of the previous set  $P_i$  and  $H_i$  leading to the activation of  $S_i$  in the Boolean network (such that the number of  $P_i$  minus the number of  $H_i$  is greater than  $\theta_i$ ) defines a reaction in the reaction system. Indeed, each possible activation in  $(M, E)$  is simulated by the enabling of a corresponding reaction having  $S_i$  as *product*, the set  $P_i$  as *reactants* and the set  $I_i = In(S_i) \setminus H_i$  as *inhibitors*. Note that by Rule 1 such reactions may belong to the reaction system since (i)  $P_i$  is a subset of the elements which may activate  $S_i$ ; (ii)  $I_i$  are a subset of the elements which may inhibit it; (iii) we know that the number of  $P_i$  minus those of  $In(S_i) \setminus I_i = H_i$  must be strictly greater than  $\theta_i$ . Note that if  $P_i$  minus those of  $In(S_i) \setminus I_i = H_i$  is equal to  $\theta_i + 1$  the reaction is actually defined by Rule 1 and added to the reaction system, otherwise, it is possible to see that such reaction would be redundant and therefore is not added. Indeed, the equality requirement in Rule 1 guarantees that only minimal combinations of reactants and inhibitors are considered (see the example below for an intuition).

Rule 2 applies only in case of self activating nodes by adding a new rule that models the self activation. In this

case, the reaction which simulates the self activating behaviour has  $S_i$  as *reactant* and also as *product*.

As an example, consider the translation of the threshold Boolean network  $(M, E)$  presented in Fig. 6. By assuming that all threshold parameters are 0, we obtain the closed reaction system  $\mathcal{RS}((M, E)) = (M, A)$  with reactions  $A$  defined as follows:

$$\begin{array}{llll}
 (\{D\}, \{C\}, \{A\}) & (\{D\}, \{C\}, \{B\}) & (\{A\}, \{C\}, \{A\}) & (\{C\}, \{D\}, \{C\}) \\
 (\{A, D\}, \emptyset, \{B\}) & (\{B\}, \{D\}, \{C\}) & (\{A, D\}, \{A\}, \{A\}) & (\{B, C\}, \{A\}, \{C\}) \\
 (\{A\}, \{C\}, \{B\}) & (\{C\}, \emptyset, \{D\}) & & 
 \end{array}$$

The reactions on the first two columns on the left are obtained by applying Rule 1, while those in the two columns on the right by applying Rule 2. Note that the requirements of Rule 1 guarantee that only minimal combinations of reactants and inhibitors are considered. For instance, a reaction such as  $(\{A, D\}, \{C\}, \{B\})$  is not present in the set of reactions because it is subsumed by  $(\{A, D\}, \emptyset, \{B\})$  (second line on the left), since the latter reaction can be applied regardless the presence of  $C$ .

The rules obtained from the translation can then be used to execute the reaction system that simulate the behaviour of the Boolean network. Thus, the following evolution can be observed:

$$\{D\} \rightarrow_A \{A, B\} \rightarrow_A \{A, B, C\} \rightarrow_A \{C, D\} \rightarrow_A \{D\}$$

As expected, the steps of the execution of the reaction system “mimic” the evolution of the Boolean network as described in Section 4.2.

In [8], the encoding of the yeast cell cycle Boolean network of Fig. 7 leads to a reaction system of 52 reactions that are used to make several virtual experiments of gene knock-out. In [7] this approach pushed forward by applying well-known methodologies and tools to compute predictors and ancestor formulas to detect alternative initial configurations for the yeast cell cycle and the genes that are necessary for such a cycle to be performed.

## 6 Conclusions

Gene regulatory networks play a crucial role in many biological processes, as cell differentiation, metabolism, the cell cycle, and signal transduction. Understanding the fundamental properties governing these mechanisms is one of the main goals of systems biology, and can be achieved by applying computational tools.

In this paper, we have provided a survey of computational approaches to GRN modeling and analysis. After an overview of the background notions, we have described how the quantitative methods, such as the ODEs and stochastic

simulations, can give an exact description of the gene regulation dynamics. However, the main weakness of the quantitative approaches is that they usually require a large number of parameters that are arduous to collect, such as kinetic constants and concentration of substances. Therefore, in the second part, we have described some more abstract qualitative models, such as Boolean networks, Petri nets, P systems, and reaction systems, which enable a basic understanding of the different functionalities of a system under different conditions. These formalisms are characterized by a lower level of resolution, and, for this reason, they are often preferable because of their simplicity and low computational complexity. Nevertheless, as discussed throughout this manuscript, they provide an approximate description of the modelled system dynamics, useful for understanding some internal mechanisms, that often require validation against real data.

**Acknowledgments** Open access funding provided by Università di Pisa within the CRUI-CARE Agreement.

## Compliance with ethical standards

**Conflict of interest** The authors declare that they have no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

1. Akutsu, T., Miyano, S., & Kuhara, S. (1999). *Algorithms for inferring qualitative models of biological networks*. *Biocomputing 200* (pp. 293–304). Singapore: World Scientific.
2. Akutsu, T., Miyano, S., & Kuhara, S. (1999). *Identification of genetic networks from a small number of gene expression patterns under the boolean network model*. *Biocomputing 99* (pp. 17–28). Singapore: World Scientific.
3. Albert, R., & Othmer, H. G. (2003). The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in *Drosophila melanogaster*. *Journal of theoretical biology*, *223*(1), 1–18.
4. Alur, R., Courcoubetis, C., Henzinger, T. A., & Ho, P. H. (1993). *Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems*. *Hybrid systems* (pp. 209–229). Berlin, Heidelberg: Springer.
5. Barbacari, N., Profir, A., & Zelinschi, C. (2005). *Gene regulatory network modelling by means of membrane systems*. *Proc WMC6* (pp. 162–178). New Jersey: Citeseer.
6. Barbuti, R., Bernasconi, A., Gori, R., & Milazzo, P. (2018). Computing preimages and ancestors in reaction systems. In: *International Conference on Theory and Practice of Natural Computing*. (pp. 23–35). Cham: Springer
7. Barbuti, R., Bernasconi, A., Gori, R., & Milazzo, P. (2020). Characterization and computation of ancestors in reaction systems. *Soft Computing*, Springer, in press.
8. Barbuti, R., Bove, P., Gori, R., Gruska, D., Levi, F., & Milazzo, P. (2019). Encoding threshold boolean networks into reaction systems for the analysis of gene regulatory networks. *Fundamenta Informaticae*, *20*, 1–20.
9. Barbuti, R., Bove, P., Gori, R., Levi, F., & Milazzo, P. (2018). Simulating gene regulatory networks using reaction systems. *CS&P*
10. Barbuti, R., Gori, R., Levi, F., & Milazzo, P. (2016). Investigating dynamic causalities in reaction systems. *Theoretical Computer Science*, *623*, 114–145.
11. Barbuti, R., Gori, R., Levi, F., & Milazzo, P. (2016). Specialized predictor for reaction systems with context properties. *Fundamenta Informaticae*, *147*(2–3), 173–191.
12. Barbuti, R., Gori, R., & Milazzo, P. Encoding boolean networks into reaction systems for investigating causal dependencies in gene regulation. *Theoretical Computer Science* (To appear)
13. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P., Pardini, G., & Tesei, L. (2011). Spatial p systems. *Natural Computing*, *10*(1), 3–16.
14. Barnes, D. J., & Chu, D. (2010). *Introduction to Modeling for Biosciences* (1st ed.). Incorporated: Springer Publishing Company.
15. Bernardini, F., Gheorghe, M., & Krasnogor, N. (2007). Quorum sensing p systems. *Theoretical Computer Science*, *371*(1–2), 20–33.
16. Bernardini, F., Gheorghe, M., Krasnogor, N., Muniyandi, R.C., Pérez-Jiménez, M.J., & Romero-Campero, F.J. (2005). On p systems as a modelling tool for biological systems. In: *International Workshop on Membrane Computing*. (pp. 114–133). Springer
17. Bilke, S., & Sjunnesson, F. (2001). Stability of the kauffman model. *Physical Review E*, *65*(1), 016129.
18. Blakes, J., Twycross, J., Romero-Campero, F. J., & Krasnogor, N. (2011). The infobiotics workbench: an integrated in silico modelling platform for systems and synthetic biology. *Bioinformatics*, *27*(23), 3323–3324.
19. Bornholdt, S. (2008). Boolean network models of cellular regulation: prospects and limitations. *Journal of the Royal Society Interface*, *5*(suppl-1), S85–S94.
20. Bottoni, P., Martín-Vide, C., Păun, G., & Rozenberg, G. (2002). Membrane systems with promoters/inhibitors. *Acta Informatica*, *38*(10), 695–720.
21. Brijder, R., Ehrenfeucht, A., Main, M., & Rozenberg, G. (2011). A tour of reaction systems. *International Journal of Foundations of Computer Science*, *22*(07), 1499–1517.
22. Bruex, A., Kainkaryam, R. M., Wieckowski, Y., Kang, Y. H., Bernhardt, C., Xia, Y., et al. (2012). A gene regulatory network for root epidermis cell differentiation in arabidopsis. *PLoS Genet*, *8*(1), e1002446.
23. Busi, N. (2002). Analysis issues in petri nets with inhibitor arcs. *Theoretical Computer Science*, *275*(1–2), 127–177.
24. Chaouiya, C., Remy, E., Mossé, B., & Thieffry, D. (2003). *Qualitative analysis of regulatory graphs: a computational tool based on a discrete formal framework Positive systems* (pp. 119–126). Berlin: Springer.
25. Chaouiya, C., Remy, E., Ruet, P., & Thieffry, D. (2004). Qualitative modelling of genetic networks: From logical regulatory graphs to standard petri nets. In: *International Conference on Application and Theory of Petri Nets*. (pp. 137–156). Springer

26. Chaouiya, C., Remy, E., & Thieffry, D. (2008). Petri net modelling of biological regulatory networks. *Journal of Discrete Algorithms*, 6(2), 165–177.
27. Chaves, M., Albert, R., & Sontag, E. D. (2005). Robustness and fragility of boolean models for genetic regulatory networks. *Journal of theoretical biology*, 235(3), 431–449.
28. Chaves, M., Sontag, E. D., & Albert, R. (2006). Methods of robustness analysis for boolean models of gene control networks. *IEEE Proceedings-Systems Biology*, 153(4), 154–167.
29. Chen, K. C., Calzone, L., Csikasz-Nagy, A., Cross, F. R., Novak, B., & Tyson, J. J. (2004). Integrative analysis of cell cycle control in budding yeast. *Molecular biology of the cell*, 15(8), 3841–3862.
30. Ciocchetta, F., & Guerriero, M. L. (2009). Modelling biological compartments in bio-pepa. *Electronic Notes in Theoretical Computer Science*, 227, 77–95.
31. Clancy, S., & Brown, W. (2008). Translation: DNA to mRNA to protein. *Nature Education*, 1(1), 101.
32. Clough, E., & Barrett, T. (2016). *The gene expression omnibus database. Statistical genomics* (pp. 93–110). New Jersey: Springer.
33. Corolli, L., Maj, C., Marini, F., Besozzi, D., & Mauri, G. (2012). An excursion in reaction systems: from computer science to biology. *Theoretical computer science*, 454, 95–108.
34. Cussat-Blanc, S., Harrington, K., & Banzhaf, W. (2019). Artificial gene regulatory networks—a review. *Artificial Life*, 24(4), 296–328.
35. Danos, V., Feret, J., Fontana, W., & Krivine, J. (2008). Abstract interpretation of cellular signalling networks. In: International Workshop on Verification, Model Checking, and Abstract Interpretation. (pp. 83–97). Springer
36. Delgado, F. M., & Gómez-Vela, F. (2019). Computational methods for gene regulatory networks reconstruction and analysis: a review. *Artificial Intelligence in Medicine*, 95, 133–145.
37. Deuffhard, P., & Röblitz, S. (2015). *A guide to numerical modeling in systems biology* (Vol. 12). Cham: Springer.
38. Ehrenfeucht, A., & Rozenberg, G. (2007). Reaction systems. *Fundamenta Informaticae*, 75(1–4), 263–280.
39. Emmert-Streib, F., Dehmer, M., & Haibe-Kains, B. (2014). Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Frontiers in cell and developmental biology*, 2, 38.
40. Fauré, A., Naldi, A., Chaouiya, C., & Thieffry, D. (2006). Dynamical analysis of a generic boolean model for the control of the mammalian cell cycle. *Bioinformatics*, 22(14), e124–e131.
41. Feinberg, M. (2019). *Foundations of chemical reaction network theory* (Vol. 202). Cham: Springer.
42. Frisco, P., Gheorghe, M., & Pérez-Jiménez, M. J. (2014). *Applications of membrane computing in systems and synthetic biology*. Cham: Springer.
43. Garg, A., Di Cara, A., Xenarios, I., Mendoza, L., & De Micheli, G. (2008). Synchronous versus asynchronous modeling of gene regulatory networks. *Bioinformatics*, 24(17), 1917–1925.
44. Gilbert, D., & Heiner, M. (2006). From petri nets to differential equations—an integrative approach for biochemical network analysis. In: International Conference on Application and Theory of Petri Nets. (pp. 181–200). Springer
45. Gillespie, D. T. (1977). Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25), 2340–2361.
46. Gillespie, D. T. (2007). Stochastic simulation of chemical kinetics. *Annu Rev Phys Chem*, 58, 35–55.
47. Goss, P. J., & Peccoud, J. (1998). Quantitative modeling of stochastic systems in molecular biology by using stochastic petri nets. *Proceedings of the National Academy of Sciences*, 95(12), 6750–6755.
48. Goss, P. J., & Peccoud, J. (1999). *Analysis of the stabilizing effect of rom on the genetic network controlling ColE1 plasmid replication. Biocomputing* (pp. 65–76). Singapore: World Scientific.
49. Hardy, S., & Robillard, P. N. (2004). Modeling and simulation of molecular biology systems using petri nets: modeling goals of various approaches. *Journal of Bioinformatics and Computational Biology*, 2(04), 619–637.
50. Harvey, I., & Bossomaier, T. (1997). Time out of joint: Attractors in asynchronous. In: Fourth European Conference on Artificial Life, vol. 4, (p. 67). Cambridge: MIT Press
51. Hecker, M., Lambeck, S., Toepfer, S., Van Someren, E., & Guthke, R. (2009). Gene regulatory network inference: data integration in dynamic models—a review. *Biosystems*, 96(1), 86–103.
52. Henzinger, T. A. (2000). *The theory of hybrid automata. Verification of digital and hybrid systems* (pp. 265–292). Cham: Springer.
53. Hinze, T., Hayat, S., Lenser, T., Matsumaru, N., & Dittrich, P. (2007). Hill kinetics meets p systems: a case study on gene regulatory networks as computing agents in silico and in vivo. In: International Workshop on Membrane Computing. (pp. 320–335). Springer
54. Karlebach, G., & Shamir, R. (2008). Modelling and analysis of gene regulatory networks. *Nature Reviews Molecular Cell Biology*, 9(10), 770–780.
55. Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3), 437–467.
56. Konur, S., Ladroue, C., Fellermann, H., Sanassy, D., Mierla, L., Ipate, F., Kalvala, S., Gheorghe, M., & Krasnogor, N. (2014). Modeling and analysis of genetic boolean gates using infobiotics workbench. *Verification of Engineered Molecular Devices and Programs* (pp. 26–37).
57. Li, F., Long, T., Lu, Y., Ouyang, Q., & Tang, C. (2004). The yeast cell-cycle network is robustly designed. *Proceedings of the National Academy of Sciences*, 101(14), 4781–4786.
58. Li, P., Zhang, C., Perkins, E. J., Gong, P., & Deng, Y. (2007). Comparison of probabilistic boolean network and dynamic bayesian network approaches for inferring gene regulatory networks. *BMC bioinformatics*, vol. 8, p. S13. Springer
59. Li, S., Assmann, S. M., & Albert, R. (2006). Predicting essential components of signal transduction networks: a dynamic model of guard cell abscisic acid signaling. *PLoS Biology*, 4(10), e312.
60. Li, X., Omotere, O., Qian, L., & Dougherty, E. R. (2017). Review of stochastic hybrid systems with applications in biological systems modeling and analysis. *EURASIP Journal on Bioinformatics and Systems Biology*, 2017(1), 8.
61. Loew, L. M., & Schaff, J. C. (2001). The virtual cell: a software environment for computational cell biology. *TRENDS in Biotechnology*, 19(10), 401–406.
62. Longabaugh, W. J. (2012). *Biotapestry: a tool to visualize the dynamic properties of gene regulatory networks. Gene regulatory networks* (pp. 359–394). New Jersey: Springer.
63. Ma’ayan, A. (2017). Complex systems biology. *Journal of the Royal Society Interface*, 14(134), 20170391.
64. Maheshri, N., & O’Shea, E. K. (2007). Living with noisy genes: how cells function reliably with inherent variability in gene expression. *Annual Review of Biophysics and Biomolecular Structure*, 36, 413–414.
65. Matsuno, H., Doi, A., Nagasaki, M., & Miyano, S. (1999). Hybrid petri net representation of gene regulatory network. Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing (pp. 341–52).
66. Mendenhall, M. D., & Hodge, A. E. (1998). Regulation of cdc28 cyclin-dependent protein kinase activity during the cell cycle of the yeast *Saccharomyces cerevisiae*. *Microbiology and Molecular Biology Reviews*, 62(4), 1191–1243.



67. Mendoza, L., Thieffry, D., & Álvarez-Buylla, E. R. (1999). Genetic control of flower morphogenesis in *Arabidopsis thaliana*: a logical analysis. *Bioinformatics*, *15*(7–8), 593–606.
68. Mendoza, L., & Xenarios, I. (2006). A method for the generation of standardized qualitative dynamical systems of regulatory networks. *Theoretical Biology and Medical Modelling*, *3*(1), 13.
69. Moore, J. H., Boczko, E. M., & Summar, M. L. (2005). Connecting the dots between genes, biochemistry, and disease susceptibility: systems biology modeling in human genetics. *Molecular Genetics and Metabolism*, *84*(2), 104–111.
70. Murata, T. (1989). Petri nets: properties, analysis and applications. *Proceedings of the IEEE*, *77*(4), 541–580.
71. Naldi, A., Remy, E., Thieffry, D., & Chaouiya, C. (2011). Dynamically consistent reduction of logical regulatory graphs. *Theoretical Computer Science*, *412*(21), 2207–2218.
72. Nasti, L. (2020). Verification of robustness property in chemical reaction networks. Ph.D. thesis, Ph. D. Thesis, University of Pisa
73. Papatheodorou, I., Moreno, P., Manning, J., Fuentes, A. M. P., George, N., Fexova, S., et al. (2020). Expression atlas update: from tissues to single cells. *Nucleic acids research*, *48*(D1), D77–D83.
74. Papin, J. A., Hunter, T., Palsson, B. O., & Subramaniam, S. (2005). Reconstruction of cellular signalling networks and analysis of their properties. *Nature Reviews Molecular Cell Biology*, *6*(2), 99–111.
75. Păun, G. (2000). Computing with membranes. *Journal of Computer and System Sciences*, *61*(1), 108–143.
76. Păun, G., & Romero-Campero, F.J. (2008). Membrane computing as a modeling framework. cellular systems case studies. In: International School on Formal Methods for the Design of Computer, Communication and Software Systems. (pp. 168–214). Springer
77. Pérez Jiménez, M.d.J., & Romero Campero, F.J. (2005). Modeling egfr signalling cascade using continuous membrane systems. In: Third International Workshop on Computational Methods in Systems Biology, CMSB 2005 (2005), (p 118–129).
78. Pérez-Jiménez, M. J., & Romero-Campero, F. J. (2006). *P systems, a new computational modelling tool for systems biology*. *Transactions on computational systems biology VI* (pp. 176–197). Cham: Springer.
79. Pescini, D., Besozzi, D., Mauri, G., & Zandron, C. (2006). Dynamical probabilistic p systems. *International Journal of Foundations of Computer Science*, *17*(01), 183–204.
80. Peter, I. S. (2019). *Methods for the experimental and computational analysis of gene regulatory networks in sea urchins*. *Methods in cell biology* (Vol. 151, pp. 89–113). Amsterdam: Elsevier.
81. Peter, I. S., & Davidson, E. H. (2015). *Genomic control process: development and evolution*. Cambridge: Academic Press.
82. Petri, C.A. (1962). Fundamentals of a theory of asynchronous information flow. In: Information Processing, Proceedings of the 2nd IFIP Congress 1962, Munich, Germany, August 27–September 1, 1962. (pp. 386–390). North-Holland
83. Pinney, J. W., Westhead, D. R., McConkey, G. A., et al. (2003). Petri net representations in systems biology. *Biochemical Society Transactions*, *31*(6), 1513–1515.
84. Polynikis, A., Hogan, S., & di Bernardo, M. (2009). Comparing different ode modelling approaches for gene regulatory networks. *Journal of Theoretical Biology*, *261*(4), 511–530.
85. Priami, C., & Quaglia, P. (2004). Beta binders for biological interactions. In: International Conference on Computational Methods in Systems Biology. (pp. 20–33). Springer
86. Profir, A., Gutuleac, E., & Boian, E. (2006). P transducer model of allelic gene network regulation. In: Proceedings of the 8-th International Conference on DAS, pp. 25–27
87. Regev, A., Panina, E. M., Silverman, W., Cardelli, L., & Shapiro, E. (2004). Bioambients: an abstraction for biological compartments. *Theoretical Computer Science*, *325*(1), 141–167.
88. Regev, A., Silverman, W., & Shapiro, E. (2000). *Representation and simulation of biochemical processes using the  $\pi$ -calculus process algebra*. *Biocomputing 2001* (pp. 459–470). Singapore: World Scientific.
89. Reisig, W. (2012). *Petri nets: an introduction* (Vol. 4). Berlin: Springer Science & Business Media.
90. Reisig, W., & Rozenberg, G. (1998). *Lectures on Petri nets i: basic models: advances in Petri nets*. Berlin: Springer Science & Business Media.
91. Romero-Campero, F. J., & Pérez-Jiménez, M. J. (2008). Modeling gene expression control using p systems: the lac operon, a case study. *BioSystems*, *91*(3), 438–457.
92. Saadatpour, A., Albert, I., & Albert, R. (2010). Attractor analysis of asynchronous boolean models of signal transduction networks. *Journal of Theoretical Biology*, *266*(4), 641–656.
93. Saadatpour, A., Wang, R. S., Liao, A., Liu, X., Loughran, T. P., Albert, I., et al. (2011). Dynamical and structural analysis of a t cell survival network identifies novel candidate therapeutic targets for large granular lymphocyte leukemia. *PLoS Computational Biology*, *7*(11), e1002267.
94. Schlitt, T., & Brazma, A. (2007). Current approaches to gene regulatory network modelling. *BMC Bioinformatics*, *8*(S6), S9.
95. Shmulevich, I., Dougherty, E. R., Kim, S., & Zhang, W. (2002). Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, *18*(2), 261–274.
96. Shmulevich, I., Dougherty, E. R., & Zhang, W. (2002). From boolean to probabilistic boolean networks as models of genetic regulatory networks. *Proceedings of the IEEE*, *90*(11), 1778–1792.
97. Somogyi, R., & Sniegowski, C. A. (1996). Modeling the complexity of genetic networks: understanding multigenic and pleiotropic regulation. *Complexity*, *1*(6), 45–63.
98. Steggles, L. J., Banks, R., Shaw, O., & Wipat, A. (2007). Qualitatively modelling and analysing genetic regulatory networks: a petri net approach. *Bioinformatics*, *23*(3), 336–343.
99. Thompson, D., Regev, A., & Roy, S. (2015). Comparative analysis of gene regulatory networks: from network reconstruction to evolution. *Annual Review of Cell and Developmental Biology*, *31*, 399–428.
100. Valencia-Cabrera, L., Garcia-Quismondo, M., & Pérez-Jiménez, M.D., Su, Y., Yu, H., & Pan, L., (2013). Modeling logic gene networks by means of probabilistic dynamic p systems. *IJUC*, *9*(5–6), 445–464.
101. Vilar, J. M., Guet, C. C., & Leibler, S. (2003). Modeling network dynamics: the lac operon, a case study. *The Journal of Cell Biology*, *161*(3), 471–476.
102. Vinh, N. X., Chetty, M., Coppel, R., & Wangikar, P. P. (2012). Gene regulatory network modeling via global optimization of high-order dynamic bayesian network. *BMC Bioinformatics*, *13*(1), 131.
103. Wang, R. S., Saadatpour, A., & Albert, R. (2012). Boolean modeling in systems biology: an overview of methodology and applications. *Physical Biology*, *9*(5), 055001.
104. Wang, S., Chen, Y., Wang, Q., Li, E., Su, Y., & Meng, D. (2010). Analysis for gene networks based on logic relationships. *Journal of Systems Science and Complexity*, *23*(5), 999–1011.
105. Wilkinson, D. J. (2018). *Stochastic modelling for systems biology*. Boca Raton: CRC Press.
106. Wong, P., Gladney, S., & Keasling, J. D. (1997). Mathematical model of the lac operon: inducer exclusion, catabolite repression, and diauxic growth on glucose and lactose. *Biotechnology Progress*, *13*(2), 132–143.
107. Yildirim, N., & Kazanci, C. (2011). *Deterministic and stochastic simulation and analysis of biochemical reaction networks: the lactose operon example*. *Methods in enzymology* (Vol. 487, pp. 371–395). Amsterdam: Elsevier.

108. Zanudo, J. G., Aldana, M., & Martínez-Mekler, G. (2011). *Boolean threshold networks: virtues and limitations for biological modeling*. *Information processing and biological systems* (pp. 113–151). Berlin: Springer.
109. Zhang, R., Shah, M. V., Yang, J., Nyland, S. B., Liu, X., Yun, J. K., et al. (2008). Network model of survival signaling in large granular lymphocyte leukemia. *Proceedings of the National Academy of Sciences*, *105*(42), 16308–16313.
110. Zhao, Q. (2005). A remark on “scalar equations for synchronous boolean networks with biological applications” by c. farrow, j. heidel, j. maloney, and j. rogers. *IEEE Transactions on Neural Networks*, *16*(6), 1715–1716.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.