Logical Modeling and Analysis of Regulatory Genetic Networks in a Non Monotonic Framework

N. Mobilia[1], A. Rocca[1], S. Chorlton[2], E. Fanchon[1], and L. Trilling[1]

[1] Laboratoire TIMC-IMAG, Université de Grenoble, France
[2] Department of Medicine, McMaster University, Hamilton, Canada

Abstract. We present a constraint based declarative approach for analyzing qualitatively genetic regulatory networks (GRNs) with the discrete formalism of R. Thomas. For this purpose, we use the logic programming technology ASP (Answer Set Programming) whose related logic is non monotonic. Our aim is twofold. First, we give a formal modeling of both Thomas' GRNs and biological data like experimental behaviors and gene interactions and we evaluate the declarative approach on three real biological applications. Secondly, for taking into account both gene interaction properties which are only **generally** true and automatic inconsistency repairing, we introduce an optimized modeling which leads us to exhibit new logical expressions for the conjunction of defaults and to show that they can be applied safely to Thomas' GRNs.

Keywords: computational systems biology · gene networks · discrete modeling · AI-oriented declarative approach · non monotonic logic · Answer Set Programming

1 Introduction

Mathematical modeling and simulation tools may help to understand how complex genetic regulatory networks (GRNs), composed of many genes and their intertwined interactions, control the functioning of living systems. They provide a framework to unambiguously describe the network structure and to infer predictions of the dynamical behavior of the system.

The typical model building cycle starts with gathering existing knowledge on a biological system and formulating working hypotheses, on the basis of which a model formalism is chosen and the structure of the GRN is defined. The development of the dynamical model and its parametrization lead to an initial model, whose predictions are confronted with experimental data. This often reveals inconsistencies, and calls for a revision of the structure of the GRN and/or the parameter values of the model. The process is repeated iteratively until the validation step is considered satisfactory. The generate-and-test approach underlying the above-mentioned method demands a large number of simulations to be carried out and usually leads to the formulation of a unique model consistent with biological data. In this paper, we adopt an alternative method for the systematic construction and analysis of models of GRNs by means of an Artificial Intelligence oriented *declarative* approach. The models are developed using the formalism of R. Thomas [15], which offers an appropriate discrete description of GRNs, as most available data on regulatory interactions are qualitative. Instead of instantiating the models as in classical modeling approaches, all possible knowledge on the network

structure and its dynamics (e.g., existence of cycles or stationary states, response of the network to environmental or genetic perturbations) is formulated in the form of constraints, i.e. logical formulae. Without resorting to numerous simulations, the compatibility of the network structure with the biological constraints is determined and an *intensional* (implicit) representation of the set of consistent models is returned, in case all the constraints are satisfied. This is well suited for biological data which are often incomplete. Futhermore, in case of inconsistency, an automatic repairing can be applied. Also, for the profit of biologists, all properties, expressed in a predefined language, that are common to all consistent models can be deduced [8].

In this paper, we use for that purpose the logic programming technology ASP [2] which is based on a non monotonic logic defined with *stable* models. The aim of this paper is to show the benefits provided by ASP for the declarative approach of Thomas' GRNs. We pay a special attention to modeling methods to take advantages of the ASP non monotonic feature for tackling potential inconsistencies and for expressing gene interaction rules that are **generally** true.

The paper is organized as follows. ASP is introduced in Section 2 and the logical specifications of Thomas' GRNs together with biological data in Section 3. Then, three illustrating applications are presented in Section 4. Finally an optimized modeling exemplifying the non monotonic aspects of ASP is described in Section 5.

2 Answer Set Programming (ASP)

Here is a short presentation based on [11] which proposes the *gringo* language for expressing ASP programs.

A logical ASP program is a finite set of rules:
$a_0 \leftarrow a_1, \ldots, a_m, not\ a_{m+1}, \ldots, not\ a_n.$
where $0 \leq m \leq n$ and $\forall i \mid 0 \leq i \leq n,\ a_i$ is an atom. For any rule r, $head(r) = a_0$ is the head of the rule, and $body(r) = \{a_1, \ldots, a_m, not\ a_{m+1}, \ldots, not\ a_n\}$ is the body of the rule. If $head(r)$ is empty, r is called an *integrity constraint*. If $body(r)$ is empty, r is a *fact*.

Let A be the set of atoms, $body^+(r) = \{a \in A \mid a \in body(r)\}$ and $body^-(r) = \{a \in A \mid not\ a \in body(r)\}$. A set $X \subseteq A$ is an *answer set* (AS) or stable model of a program P if X is the minimal model [3] [4] of the *reduct*
$P^X = \{head(r) \leftarrow body^+(r) \mid r \in P, body^-(r) \cap X = \varnothing\}$.

Example: let E be the following ASP program where $\leftarrow$ is represented by `:-` :

```
a :- not b,c.
c.
```

Let $X = \{\mathtt{a}, \mathtt{c}\}$. The corresponding reduct is $E^X = \{\mathtt{a} \leftarrow \mathtt{c}, \mathtt{c}\}$ and its minimal model is $\{\mathtt{a}, \mathtt{c}\}$. Then X is an AS of E.
Let $X' = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$. The corresponding reduct is $E^{X'} = \{\mathtt{c}\}$, and its minimal model is $\{\mathtt{c}\}$. Then X' is not an AS of E.

The first rule of this example is typical of a *default* rule [5]. It expresses that generally (in the absence of knowledge on `b`) `a` is implied by `c`. But if `b` holds because

[3] A logical model is minimal when removing an atom from it cannot provide a model. A reduct has a unique minimal model.

[4] It is important to note that ASs also have been shown to be minimal (see Section 5.2).

of additional knowledge, this rule is no longer applicable. For instance, by addition of the fact `b.` to this example, we get the AS {`b,c`} exemplifying the non monotonic character of ASP: `a` does not hold any more, without leading to an inconsistency.

Rules in the *gringo* language are extended for accepting heads which are disjunctions of literals (exclusive unless both literals are proven) using the operator `|`. Furthermore, *gringo* provides logical variables and functional terms, in a limited way (so that the program can be transformed in an equivalent finite propositional one). It also provides cardinality constraints on the number of true literals. If we impose the constraint `u{l_1,...,l_n}v` we obtain only models such that the number of true literals `l_i` is bigger than (or equal to) `u` (`0` by default) and smaller than (or equal to) `v` (`n` by default). Moreover, this formalism allows the expression of conditional enumerations of literals through the symbol "`:`", as conjunctions (resp. disjunctions) in the body (resp. head) of a rule. For example, in the following program:

```
dom(0). dom(1).     all_true :- p(X) : dom(X).
q(X) : dom(X) :- one_of.
at_least_one_true :- 1{p(X) : dom(X)}.
```

the first line expresses that `all_true` is deduced if `p(0)` and `p(1)` hold. This line is, therefore, equivalent to the rule: `all_true :- p(0), p(1).` and the second line equivalent to `q(0)| q(1) :- one_of.` The third line expresses that `at_least_one_true` is deduced if a least one among `p(0)` and `p(1)` holds.

Para-logic operators are also provided for maximizing or minimizing the number of atoms true among a set of atoms. Asserting `#maximize{f_1,...,f_n}` will produce only models with the highest number of `f_i` true.

The solver [11] we use proceeds in two steps to compute the ASs of a program P. First, a "grounder" substitutes the variables of the program by terms without free variables, and consequently produces a propositional program $\mathbf{P}$ corresponding to P. In the second step, a solver computes the ASs of $\mathbf{P}$. This motivates the programmer to reduce as far as possible the number of resulting Boolean variables and rules subject to a big expansion (see Section 5.1).

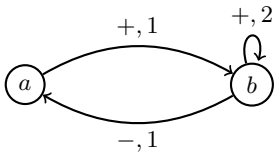
3 Thomas' GRNs and the declarative approach

In Section 3.1, we specify Thomas' GRNs. Biological constraints and typical queries for constructing and analyzing models of GRNs are described in Section 3.2.

3.1 Thomas' GRN specification

A common representation of a GRN is to view it as an *interaction graph*, where nodes represent genes and arrows represent interactions between genes. An arrow $j \rightarrow i$ is labeled with the sign of the regulatory influence (to indicate whether the gene i is activated or inhibited by the product J of gene j), and with the index r of the threshold concentration θ_j^r above which the protein J controls the expression of gene i. A simple example of interaction graph for a two-gene network is shown in Fig. 1(a).

The dynamic behavior of a GRN is represented in terms of an oriented graph called *state transition graph*, where each node represents a specific *state* of the system and the arrows represent *transitions* between a state and its possible successors. A state S of a network of n genes is represented by a vector of protein concentrations: $S = [x_1, ..., x_n]$. The concentrations take discrete values, each one representing an interval



(a)

$$
\begin{aligned}
X_a = {} & K_a * s^-(x_b, \theta_b^1) + \\
& K_a^b * s^+(x_b, \theta_b^1) \\
X_b = {} & K_b * s^-(x_a, \theta_a^1)\, s^-(x_b, \theta_b^2) + \\
& K_b^a * s^+(x_a, \theta_a^1)\, s^-(x_b, \theta_b^2) + \\
& K_b^b * s^-(x_a, \theta_a^1)\, s^+(x_b, \theta_b^2) + \\
& K_b^{ab} * s^+(x_a, \theta_a^1)\, s^+(x_b, \theta_b^2)
\end{aligned}
$$

(b)

Fig. 1. (a) Interaction graph corresponding to a GRN of two genes. The product of gene a stimulates the expression of gene b, while the product of gene b inhibits expression of a. In addition, b activates its own expression. The label $-, 1$ from gene b to a means that b inhibits a expression when b is above its threshold θ_b^1. (b) Focal equations relating a state $[x_a, x_b]$ and its focal state $[X_a, X_b]$.

between two consecutive thresholds. For instance, $x_j = 0$ indicates a concentration of protein J lower than the lowest threshold of J, say θ_j^m, whereas $x_j = 1$ means that the concentration of J is lower than $\theta_j^m + 1$ and higher than (or equal to) θ_j^m.

A specific attractor value called *focal state*, $[X_1, ..., X_n]$, is associated to a given state S. It represents the expression levels toward which the genes tend to evolve (see precise definition with *focal equations* below). A successor $S' = [x'_1, ..., x'_n]$ of S in the graph is deduced from S by comparing the value of each variable x_i with that of its focal state. The transition of S to S' is assumed to be asynchronous, in the sense that at most one variable x_i is updated at a time. If the variable x_i is updated, the formal relationship between these states is expressed as follows: $x'_i = x_i + 1$ if $X_i > x_i$ and $x'_i = x_i - 1$ if $X_i < x_i$. If no logical variable x_i is updated then the focal state of S is equal to S and S is its own successor: in that case S is said to be *steady* (or stationary).

The focal state value X_i of gene i depends on the state S of the network and in particular, on a set of conditions regarding the presence or absence of activators and inhibitors of gene i. For the simple example in Fig. 1, the focal state value X_a of gene a depends on the influence of B (the product of b) on a, that is, if the concentration of B is below ($x_b = 0$) or above its first threshold value. Such interactions are expressed by means of products of step functions of the form: $s^+(x_j, \theta_j^r) = 1$ if $x_j \geq \theta_j^r$ else 0, $s^-(x_j, \theta_j^r) = 1$ if $x_j < \theta_j^r$ else 0.

We will call *cellular context* of gene i any set of states which are equivalent with respect to i for regulation purpose. For example, if i is influenced by only one gene j associated to the threshold θ_j^1, there are only two possible cellular contexts for i, depending on whether x_j is below or above θ_j^1. If i is the target of two interactions, four contexts have to be distinguished for i. More formally, let $[(j_1, \theta_{j_1}^{r_1}), ..., (j_p, \theta_{j_p}^{r_p})]$ be the ordered list of interactions acting on gene i. A cellular context for i is represented as a product $c_i(\sigma) = s^{\sigma_1}(x_{j_1}, \theta_{j_1}^{r_1}) * ... * s^{\sigma_p}(x_{j_p}, \theta_{j_p}^{r_p})$ defining a set of conditions, where $\sigma \in E^i = \{(\sigma_1, ..., \sigma_p) | k : 1..p, \sigma_k \in \{+, -\}\}$. In other words, a cellular context is identified by a p-tuple σ of signs, which specify the position of the region with respect to the p thresholds belonging to the interaction set. From the above definition it follows that the set of cellular contexts with respect to any gene i constitutes a partition of the state space. For the example of Fig. 1, let $[(a, \theta_a^1), (b, \theta_b^2)]$ be the list of interactions acting on b, then the cellular context $c_b((+, -))$ is $s^+(a, \theta_a^1) * s^-(b, \theta_b^2)$.

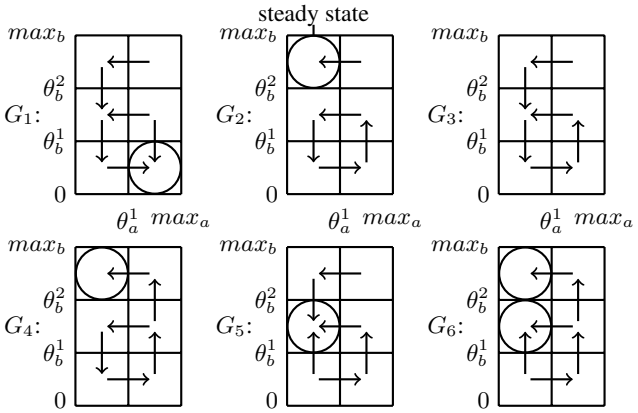


Fig. 2. Transition graphs G_1, ..., G_6 satisfying all the observability and additivity constraints associated to the example in Fig. 1, with $\theta_b^1 < \theta_b^2$. Arrows represent possible transitions between states represented by boxes. Each graph corresponds to a specific set of instantiated kinetic parameters.

Each cellular context $c_i(\sigma)$ is associated to a *logical kinetic parameter* defining the focal value of gene i when the network state belongs to that context. We will denote such parameters by $K_i^{l(\sigma)}$, where $l(\sigma)$ is the set $\{j_k | k : 1..p, \sigma_k = +\}$. In this way, the cellular context associated to a logical parameter appears in its notation. For the simple example in Fig. 1, the logical parameters of gene b are: K_b, K_b^b, K_b^a and K_b^{ab}. They respectively are the focal value of b: when the concentrations of A and B are under their thresholds, when the concentration of B only is above its threshold, when the concentration of A only is above its threshold, and when both concentrations are above their thresholds.

Given the cellular contexts defined above and their associated logical parameters, X_i is specified by the following focal equation: $X_i = \sum_{\sigma \in E^i} K_i^{l(\sigma)} * c_i(\sigma)$.

In the declarative approach, some logical parameters may not be instantiated (i.e. unknown). When all these parameters are instantiated, the focal equations define a unique instantiated model. The focal equations corresponding to the simple network example are shown in Fig. 1(b). For the following values of logical parameters: $K_a = 1$, $K_a^b = 0$, $K_b = 0$, $K_b^b = 1$, $K_b^a = 0$, and $K_b^{ab} = 2$, we obtain the transition graph G_1 in Fig. 2. The state $[1, 1]$ belongs to the cellular context $c_a((+))$ and to the cellular context $c_b((+, -))$. The focal state of $[1, 1]$ is therefore $\left[K_a^b, K_b^a\right] = [0, 0]$. It follows that the successors of $[1, 1]$ are $[1, 0]$ and $[0, 1]$.

3.2 Biological data modeling

We focus here on interactions between genes and observed experimental behaviors of the network [5].

Behaviors Experimental behavioral data are expressed using constraints on *paths* which are successions of states. This is the case for modeling observed steady states, cycles or repairing behavior due to stress. The declarative approach presents a decisive advantage as information on these behaviours is usually incomplete; for example, there could exist a cycle for which only some concentrations of proteins are known throughout the cycle. Despite the lack of information, this approach may provide biologically meaningful properties regarding the kinetic parameters. Constraint modeling of paths is described with the predicate `species(N, V, I, P)` meaning that `V` is the expression level of the gene `N` at step `I` of the path `P`. For example, defining a steady state can be done through the predicate `statpath(P)` (the two states of the path `P` of length 2 are equal):

```
statpath(P) :- path(P), length(2,P), succeg1(N,P):node(N).
```

where `succeg1(N,P)` is true if at the two first steps of the path `P`, the concentrations of the species `N` are equal. Enforcing the existence of the steady state `ss` can then be performed with the two facts and the integrity constraint which follow:

```
path(ss). length(2,ss). :- not statpath(ss).
```

Applied to the example of Fig. 1, this gives the models corresponding to the transition graphs $\{G_1, G_2, G_4, G_5, G_6\}$ of Fig. 2.

This expressive power provides significant benefits over well-known temporal logics like Computational Tree Logic (CTL), which have been proposed to check instantiations of Thomas networks [4]. For example, a query asking for the existence of a model admitting three different steady states (without knowing them beforehand), is easy to formulate as an extension of the above rules, but cannot be expressed in CTL.

Nevertheless, CTL is useful to express biological observations, typically with $EF\varphi$ formulas applied to a state, meaning that there exists at least a path originating from this state leading to a state with property φ. In the declarative framework, one can easily enforce such formulas. For the example in Fig. 1, enforcing the existence of a path respecting the CTL formula $(a = 0 \wedge b = 0) \wedge EF(a = 0 \wedge b = 2)$ (there exists a path beginning with a state respecting $a = 0$ and $b = 0$ and reaching a state where $a = 0$ and $b = 2$) is achieved with the following rules[6]:

```
path(p). length(5,p).
:- not species(a,0,1,p). :- not species(b,0,1,p).
exist_path :- species(a,0,I,p), species(b,2,I,p).
:- not exist_path.
```

The only models satisfying this formula are G_4 and G_6 (Fig. 2).

Enforcing universal CTL properties like $AF\varphi$, meaning that from the state to which it is applied all paths lead to a state with property φ, is not easily handled (see Section 6). However, such formulae are not appropriate for transcribing a biological experiment

[5] Modeling other data like those obtained from *mutant* networks, resulting from suppression or over-expression of genes, can be found in [9].

[6] The length of `p` is set to `5` because it is the maximal length of a non looping path for this example.

in CTL because the experiments usually include only a few trials. An AF formula would be too strong: some valid models could be unduly eliminated. However, if we change the context, and search for constructing robust networks in a synthetic biological perspective, it becomes crucial, as it is in computer programming, to ensure universal properties.

Interaction signs It is mandatory in a logical framework to define rigorously the meaning of the signs "+" or "−" on edges of an interaction graph (they may be loosely interpreted in the literature). We propose here a rather general and intelligible definition in the form of conditions called *observability constraints* and *addivity constraints* (not to be confused with the ASP integrity constraints).

A "+" (resp. "−") sign on an edge targeting a gene is understood as implying the existence of a couple of states $(s1, s2)$, with $s1$ just below the edge threshold, such that 1) $s2$ differs from $s1$ only by a $+1$ change in the value of the source gene, and 2) $s2$ has a greater (resp. lower) focal value for the target gene than $s1$. One may see why the transition graph G_4 (Fig. 2) respects the "+" label associated with the edge $a \rightarrow b$ (Fig. 1). The state $[0, 1]$ is such that the value of the source node a is lower than the threshold θ_a^1 of this edge. This state has a neighbouring state $[1, 1]$, which differs only in the value of a by a change of $+1$. Furthermore, this neighbour shows a positive tendency ($K_b^a = 2$) for b, indicating a future growth in expression level, while the state $[0, 1]$ shows a negative one ($K_b = 0$).

As all states of a cellular context have the same focal state, the existence of states $(s1, s2)$ is equivalent to the existence of cellular contexts $(c1, c2)$ of the target node which have the following properties for a "+"(resp. "−") sign: all states in $c1$ below the edge threshold and 1) $c2$ differs from $c1$ only by value of the source gene greater or equal than the edge threshold and 2) the focal value of the target gene in the context $c2$ has a greater (resp. lower) value than in context $c1$. In the transition graph G_4, considering again the positive interaction $a \rightarrow b$, such a couple of cellular contexts of b is for $c1$ the cellular context where $a < \theta_a^1 \wedge b < \theta_b^2$ holds and for $c2$ where $a \geq \theta_a^1 \wedge b < \theta_b^2$ holds.

Then, observability constraints for an interaction are expressed by a union of strict inequalities between kinetic parameters of the target of this interaction, just differing by one gene. For example, the observability constraint associated to the positive interaction $a \rightarrow b$ is $(K_b < K_b^a) \vee (K_b^b < K_b^{ab})$.

Additivity constraints are considered to indicate that **generally** no inhibition (resp. activation) can exist in case of a positive (resp. negative) interaction. For example, this means that in the general case for the positive interaction $a \rightarrow b$ where there is no proven inhibition (e.g. $(K_b > K_b^a) \vee (K_b^b > K_b^{ab})$ does not hold), then the negation of this inhibition is true (e.g. $(K_b \leq K_b^a) \wedge (K_b^b \leq K_b^{ab})$ holds). Solving this issue requires this ASP predicate:

```
addit(S,N1,N) :- obs(S,N1,N), opposite_sign(S,Sp),
                 not obs(Sp,N1,N).
```

where `obs(S,N1,N)` means that if `S=p` (resp. `S=m`) then the edge `N1`→`N` is an activation (resp. inhibition) and to introduce the integrity constraint:

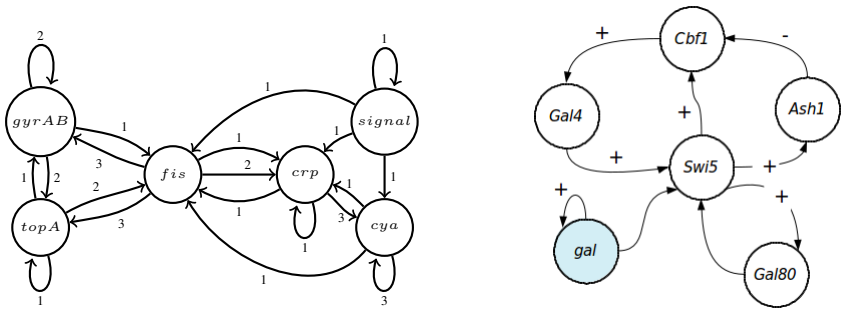
```
:- addit(S, N1, N), opposite_sign(S, Sp),
   not -obs(Sp, N1, N).
```

where `-obs(S,N1,N)` is the (usual) negation of `obs(S,N1,N)`. One should notice the default character of the rule defining `addit`. In the general case `obs(Sp,N1,N)` is not established, thus this rule is applied, with possible consequences. But if, due to the addition of new data, `obs(Sp,N1,N)` is established, there will be no inconsistency because of these consequences.

For reducing the number of ASs, and then increasing the number of properties deduced from them, a rather radical criterion (discussed in Section 5.2) can be applied by maximizing, with the para-logical operator `#maximize`, the number of `addit` atoms.

4 Applications

The three applications that are presented below illustrate the following advantages of the declarative approach: inconsistency checking and repairing, minimization of interaction and threshold numbers, and temporal series modeling.



(a) (b)

Fig. 3. (a) Interaction graph of the regulation of the carbon starvation response in *Escherichia coli* [8]. (b) Interaction graph of the IRMA network.

Carbon starvation response in *Escherichia coli* The declarative approach has been applied to the re-examination of the regulation network of the carbon starvation in *E. coli* presented in [13]. As long as environmental conditions are favourable, a population of *E. coli* bacteria grows quickly. The bacteria are in a state called exponential phase. Upon a nutritional stress due to carbon starvation, the bacteria are no longer able to maintain a fast growth rate. They enter in a stationary phase. Their response can be reversed as soon as the environmental conditions become favourable again. Modeling with the generate-and-test approach classically used for constructing GRN models led to a unique, instantiated and inconsistent model.

A declarative analysis of this network, using CLP and SAT solvers, has been presented in [8]. The network (Fig. 3 (a)) and biological observations on interactions, paths (stationary states and paths leading from the exponential phase to the stationary phase and vice-versa) and even characteristics of the shape of the DNA (*supercoiling*) were described using constraints. This analysis was resumed with an ASP implementation. We illustrate here the repairing of inconsistency.

Logical inconsistency was established. This showed rigorously the non-existence of alternative models, i.e. with a reasoning not based on the inconsistency of only one particular instantiated model. Repairing inconsistency was related to additivity constraints to the extent that they were not supported experimentally. Then repairing process proposed two solutions, that is to remove one constraint among $K^{fis}_{gyrAB} \leq K_{gyrAB}$ and $K_{topA} \leq K^{fis}_{topA}$. After biological investigations, it appeared that the first one should not be removed, but that the second could be, as it can be considered as not biologically plausible.

Computer performances stay very acceptable for solving such requests which require numerous recombination computations. For example, it is for determining the removable constraints that [8] reports the highest computer time (around 25min), with Prolog and a SAT solver loosely cooperating. This result was understandable because of the size of the solution space in this case. The same issue took 4s when solved by our ASP implementation (with a Core 2 Duo 3GHz, 4Go RAM).

Drosophila embryo gap genes network This ASP approach has been applied in [7] to the regulatory network controlling the earliest steps of *Drosophila* embryo segmentation, i.e. the gap genes and their cross-regulations, under the additional control of maternal gene products [14,12,1]. Three kinds of data were considered: 1) published molecular genetic studies enabling the identification of the main actors (seven genes), as well as the establishment or the potentiality of cross-regulatory interactions, 2) qualitative information on the spatio-temporal expression profiles of the main genes involved in the process, giving seven regions with different stable states, 3) data available on the gap gene expression profiles for seven loss-of-function mutations, affecting maternal or gap genes. On the basis of this combination of interaction and gene expression constraints, the challenge was to identify the model(s) involving all established regulatory edges, along with a minimal set of potential ones, while minimizing the number of distinct thresholds.

In a first step, the consistency of the data was proven in 3338s, using a Linux PC with an Intel Core2Duo processor at 2.4GHz and 2.9GB of memory. Then, a unique regulatory network structure was obtained in 1016s which included only 2 potential interactions (on 11). Surprisingly, from this network, there was a unique instantiation of the thresholds minimizing the number of threshold values per component (obtained in 368s). Finally, some properties concerning the kinetic parameters were deduced: 52 parameters fixed (over 72), 12 inequalities connecting a threshold and a parameter, and 36 inequalities connecting two parameters.

IRMA interaction network The IRMA (In vivo benchmarking of Reverse-engineering and Modelling Approaches) network [6] comprises five genes: Swi5, Ash1, Cbf1, Gal4 and Gal80, as well as one input (gal) and eight interactions (Fig. 3 (b)). These genes were chosen in the synthesis of the network so that different types of interactions were considered, including transcription regulation and protein-protein interaction, thereby capturing the behaviour of eukaryotic GRNs. In [6] Cantone et al. explored the dynamics of the IRMA network by measuring each gene's expression level in response to two different perturbations using qRT-PCR. In the first set of experiments, they shifted yeast cells from a glucose to galactose medium ('switch-on' experiments) and in the

second set of experiments they shifted the cells from a galactose to glucose medium ('switch-off' experiments). The presence of galactose allows for increased transcription of Swi5 and is thus 'switch-on', while the opposite is true for the 'switch-off' experiments. From these data, two temporal series, composed of averaged gene expressions over five 'switch-on' and four 'switch-off' independent experiments, have been extracted.

Finding possible models of the IRMA network respecting these time series is a challenge proposed in [3]. The network is given in such terms that the order between the kinetic parameters is known. So the issue is to find a consistent order between thresholds and these parameters and between the thresholds themselves. Time series are formalized by CTL formulas of the form: $EF(prop_1 \wedge EF(...EF(prop_n)...)$ where $n = 12$ for the switch-off experiments and $n = 10$ for the switch-on experiments. A property $prop_i$ relates to the values of the components of a state and also to the derivative signs of these components. In [3] Batt et al. propose a modeling taking into account *singular states* (states admitting for a component a threshold value) leading to more states, together with the use of the model checking tool NuSMV. They claim, reviewing their work, that they provide more precise results and efficient coding.

When applying an ASP declarative approach to this problem (not yet published), we designed the appropriate constraints for expressing that a path satisfies a time series, while remaining in the Thomas framework, i.e. without singular states. The same number of parametrizations (64) were exhibited as in [3] [7], in 139s (compared to 885s, on a similar computer), showing that, at least for this analysis, taking account of singular states was not necessary.

5 Optimized modeling and additivity constraints

Two issues arise for modeling additivity constraints: 1) escaping from a possible inconsistency that would result if these constraints would be imposed, 2) getting only the "most general" networks that is, intuitively, those which accept as many as possible additivity constraints compatible with the biological data. Recall that such a minimization of the number of resulting models leads to increasing the number of properties which can be deduced. The first issue is elegantly solved[8] with the default rule presented in Section 3.2. The second one can be solved with a naive modeling [9] which enumerates all models and uses the para-logical maximization operator of the *gringo* language as suggested in Section 3.2. However, both enumerating too many atoms and using para-logic operators are costly.

The refined modeling which follows attempts to reduce as far as possible these costs by taking advantage of the non monotonicity of ASP[9]. In Section 5.1 default rules [5] will be applied for lowering the enumeration of the models. In Section 5.2 an appropriate conjunction of defaults will be introduced for logically minimizing the number of resulting models.

[7] Parameterizations were found identical, except for two of them that were erroneous in [3].

[8] A more brutal para-logical approach with a maximization operator is proposed in [8].

[9] Note that this modeling allows also to associate additivity constraints even to edges that would not be labeled by any sign in the interaction graph but that would support nonetheless observability constraints as a result of the given behaviours.

5.1 Lowering the enumeration of `kparam` atoms

The predicate `kparam(K,Ik)` means that `K` is the value of `Ik = k(N,CC)` representing $K_N^{l(\sigma)}$ with `CC` being a term representing $l(\sigma)$. Its definition could be:
`1{kparam(K,k(N,CC)):val(N,K)}1 :- param(k(N,CC)).`
where `val(N,K)` means that `K` is a possible value of `N` and `param(Ik)` that `Ik` is a parameter identifier of `N`. Unfortunately, this definition leads to an exhaustive enumeration of all networks. To produce `kparam` atoms more savingly, the method consists to design specific rules related to the three origins of their production: observability constraints due to the interaction graph, additivity constraints and biological behaviors. All of these take advantage of the non monotonic property of ASP. Here, we will focuse on rules related to the second origin.

Producing correct couples of parameters is expressed via the rules:

```
kparam(K, Ik) :- couple_kpr(K, Ik, _, _).
kparam(K_r, Ik_r) :- couple_kpr(_, _, K_r, Ik_r).
```

The `couple_kpr` atoms due to additivity constraints are introduced by the default rule:

```
1{couple_kpr(K, Ik, K_r, Ik_r)
            : -param_obs(Sp, N1, N, K, K_r)}1
  :- neighboring_cell_cont(N1, N, Ik, Ik_r),
     addit(S, N1, N), opposite_sign(S, Sp).
```

where `neighboring_cell_cont(N1, N, Ik, Ik_r)` ensures that `Ik` and `Ik_r` identify two cellular contexts of `N` separated by the edge `N1`→`N`, `Ik_r` being `Ik` less the edge `N1`→`N`. The atom `param_obs(S, N1, N, K, K_r)` ensures that `K` and `K_r`, possible parameters of `N`, are in the right order regarding observability constraints with the sign `S` considering that `K_r` is associated to a cellular context that is the one associated to `K` less (with the above meaning) the edge `N1`→`N`. The predicate `-param_obs` is the (true) negation of `param_obs`. Note that expressing the logical conjunction (of inequalities) representing an additivity constraint requires every `couple_kpr` atoms associated to a couple of cellular contexts of `N` separated by the edge `N1`→`N`.

5.2 Conjunction of defaults and appropriate use of the para-logical maximization operator

Defining the notion of "most general" networks regarding additivity constraints raise two different questions. The first is: "What is the logical definition of the answer sets satisfying observability constraints and behaviors (e.g. paths) to be retained regarding additivity constraints ?", the second: "Among the answer sets that result from the first question, are there still some to be eliminated ?". For minimizing the number of resulting models, there are two means: a logical one, coming from the minimality of the stable models and adapted to the first question, and a para-logical one via the maximization operator possibly useful for the second question.

For a set of parameters satisfying observability constraints and behaviors, it appears natural to ask for only answer sets having additivity constraints for all edges of all species, if such an answer set exists. If not, one would like to keep only the answer sets

having additivity constraints for all edges of the species for which it is possible. For example, for the network of Fig. 1 with a behavior implying only $K_b^{ab} = 2$, there are the answer sets represented by the graphs $G_1, ..., G_6$ of Fig. 2 with additivity constraints for all edges of all species. But there are also other possible networks, for example with one edge of b with no additivity constraints. Unfortunately, the above modeling provides such undesirable networks, due to possible additivity constraints for one edge that implies the non additivity for some other edges. This is the case when having additivity constraints for the edge $a \rightarrow b$ ensured with the additional parameter values $K_b = 1$, $K_b^a = 1$ and $K_b^b = 0$ (thus $(K_b \leq K_b^a) \wedge (K_b^b \leq K_b^{ab})$ holds). These parameter values forbid additivity constraints for the edge $b \rightarrow b$ since $K_b \leq K_b^{ab}$ does not hold.

A simple program would help for illustrating this last point and for exhibiting a methodology to solve it. Let us consider the two following default rules which mimic cross influences between edges:

```
ad1_2 | ad1 :- not op_ad1.   ad2_1 | ad2 :- not op_ad2.
```

where the predicates prefixed by `ad`i represent additivity constraints on edge i and the predicates `op_ad`i represent the (rarely proven) conditions preventing additivity constraints on edge i. Influences between edges are modeled with the rules:

```
op_ad2 :- ad1_2.   op_ad1 :- ad2_1.
```

These four rules have three ASs: `{ad1,ad2}`, `{ad1_2,op_ad2}` and `{ad2_1,op_ad1}`. The challenge is to transform these rules so that we only get the AS `{ad1, ad2}`. The methodology consists firstly in introducing the rules:

```
c :- op_ad1.   c :- op_ad2.
```

so that `not c` represents the case where both `op_ad1` and `op_ad2` are unknown or false, and secondly in completing the body of each original default rules with a kind of "guard" which is a tautological term provided with a default impact power:

```
ad1_2 | ad1 :- not op_ad1, 1{c, not c}1.
ad2_1 | ad2 :- not op_ad2, 1{c, not c}1.
```

The point is that when `not c` is true then `ad1_2` (or `ad2_1`) cannot be deduced. But if the fact `op_ad1.` is added then two ASs are obtained: `{op_ad1,ad2_1}` and `{op_ad1,ad2}`.

Applied to our case, this methodology simply asks for the introduction of a guard of the following form into the body of the rule producing the additivity constraints:

```
1{not one_no_addit(N), one_no_addit(N)}1,
1{not one_no_addit, one_no_addit}1
```

where `one_no_addit(N)` means that one edge leading to the species `N` is not additive and `one_no_addit` means that one species has not all its entering edges additive.

Keeping only ASs with all possible additivity constraints may be prevented not only by behaviors but also by some parameter instantiations respecting observability constraints. Guarding in the same way the rule producing observability constraints solves this issue, in the case where no other behavior occur. This necessitates to prove that by doing so at least one AS is obtained. For this purpose, it has been necessary to show that the definition of interaction signs that we propose in Section 3.2 satisfies the following theorem: "Whatever are the interactions on a species N, one from each source, there exists an AS respecting all addivity constraints, in the absence of any other constraints on the parameters".

Meanwhile, there remain cases that generate further questions. For example, for the network of Fig. 1 with at least two stationary states, this new modeling provides nonetheless 3 ASs: one with the two edges of b being additive (graph G_6) and providing the stationary states $(0,1)$ and $(0,2)$, the two others respectively with one and not any of these edges being additive and providing the stationary states $(0,1)$ and $(1,0)$. The parameters values of these last ASs come from the stationary states ($K_b = 1$ and $K_b^a = 0$) and the observability constraints ($K_b^{ab} = 2$, $K_b^b = 0$ or $K_b^b = 1$). They are acceptable models from the "logical" point of view developed above. Consequently, discriminating some ASs among these three ASs can require para-logic standards like the one suggested in Section 3.2, i.e. the winners are those having in the whole the greatest number of additive edges, which eliminates these two possibly undesirable models.

6 Conclusion

We gave the main lines of an ASP modeling of Thomas' GRNs and illustrated the declarative approach interest with biological applications which make use of inconsistency repairing, minimization of interactions and temporal series representation. To take into account properties only generally true, we presented adequate default rules and an optimized modeling both reducing the number of used atoms (an efficient way for improving performances in computational logics), and keeping only most generally accepted models in a logical way (as opposed to a para-logical one). For this purpose, we were led to express conjunction of defaults with surprising and powerful ASP logical expressions, and to apply safely such expressions to Thomas' GRNs.

Few other teams use a declarative approach for analysing Thomas' networks. For this purpose, they use model checking tools and formalize paths with the temporal logic CTL, as in a seminal paper [4] on the subject. We showed advantages of our approach in Sections 3.2 and 4. It can be added that a constraint programming approach is well suited to avoid external processes to extract properties common to the set of consistent models, because these models are defined intensionaly as solutions of the constraints. However, as mentioned in Section 3.2, imposing (opposed to checking) in a convenient way CTL formulae of the form $AF\varphi$ with logic programming remains an issue. In [9] an acceptable solution taking advantage of the underlying non monotonicity of ASP is proposed. It based on the fact that a circular combination of rules like `a1 :- a2. a2 :- a1.` has only an empty AS while the corresponding logical formulae $a1 \Leftarrow a2.\ \ a2 \Leftarrow a1.$ have also the model $\{a1, a2\}$ (which is not minimal).

Finally, it should be noted that ASP has also been applied successfully by other teams to the modeling of biological networks (see for example [16]), but not specifically, at our knowledge, to the modeling of Thomas' GRNs. The apparently closest ASP based work is reported in [10] but it deals only with simplistic instantiated deterministic Boolean networks, thus excluding issues coming from the multi-valued non deterministic Thomas'GRNs modeling in the declarative approach perspective, yet offered by the programming logic technology. Furthermore, it does not emphasize the non monotonicity offered by ASP.

Acknowledgments. This work was supported by Microsoft Research through its PhD Scholarship Programme. We acknowledge funding by the Agence Nationale de la Recherche through the CAPMIDIA project.

References

1. F. Alves and R. Dilao. Modeling segmental patterning in drosophila: Maternal and gap genes. *J. Theor. Biol.*, 241:342–359, 2006.
2. C. Baral. *Knowledge Representation, Reasoning, and Declarative Problem Solving*. Cambridge University Press, New York, NY, USA, 2003.
3. G. Batt, M. Page, I. Cantone, G. Goessler, P. Monteiro, and H. De Jong. Efficient parameter search for qualitative models of regulatory networks using symbolic model checking. *Bioinformatics*, 26(18):i603–i610, 2010.
4. G. Bernot, J.-P. Comet, A. Richard, and J. Guespin. Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology*, 229(3):339–347, 2004.
5. P. Besnard. *An Introduction to Default Logic*. Springer, 1989.
6. I. Cantone, L. Marucci, F. Iorio, M. A. Ricci, V. Belcastro, M. Bansal, S. Santini, M. di Bernardo, D. di Bernardo, and M. P. Cosma. A yeast synthetic network for in vivo assessment of reverse-engineering and modeling approaches. *Cell*, 137(1):172 – 181, 2009.
7. F. Corblin, E. Fanchon, L. Trilling, C. Chaouiya, and D. Thieffry. Automatic inference of regulatory and dynamical properties from incomplete gene interaction and expression data. In *Proceedings of the 9th International Conference on Information Processing in Cells and Tissues*, IPCAT'12, pages 25–30, Berlin, Heidelberg, 2012. Springer-Verlag.
8. F. Corblin, S. Tripodi, É. Fanchon, D. Ropers, and L. Trilling. A declarative constraint-based method for analyzing discrete genetic regulatory networks. *Biosystems*, 98:91–104, 2009.
9. L. Farinas de Cerro and K. Inoue, editors. *Logical Modeling of Biological Systems*, pages 167–206. Wiley-ISTE, London, 2014.
10. T. Fayruzov, J. Janssen, D. Vermeir, C. Cornelis, and M. D. Cock. Modelling gene and protein regulatory networks with answer set programming. *Int. J. Data Min. Bioinformatics*, 5(2):209–229, Mar. 2011.
11. M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, and S. Thiele. *A user's guide to gringo, clasp, clingo, and iclingo (version 3.x)*, Oct 2010.
12. J. Jaeger, M. Blagov, D. Kosman, K. N. Kozlov, Manu, E. Myasnikova, S. Surkova, C. E. Vanario-Alonso, M. Samsonova, D. H. Sharp, and J. Reinitz. Dynamical analysis of regulatory interactions in the gap gene system of drosophila melanogaster. *Genetics*, 167:1721–1737, 2004.
13. D. Ropers, H. de Jong, M. Page, D. Schneider, and J. Geiselmann. Qualitative simulation of the carbon starvation response in *escherichia coli*. *Biosystems*, 84(2):124–152, 2006.
14. L. Sánchez and D. Thieffry. A logical analysis of the *Drosophila* gap-gene system. *J. Theor. Biol.*, 211:115–141, 2001.
15. R. Thomas and M. Kaufman. Multistationarity, the basis of cell differentiation and memory. II. logical analysis of regulatory networks in terms of feedback circuits. *CHAOS*, 11(1):180–195, 2001.
16. S. Videla, C. Guziolowski, F. Eduati, S. Thiele, M. Gebser, J. Nicolas, J. Saez-Rodriguez, T. Schaub, and A. Siegel. Learning boolean logic models of signaling networks with ASP. *Theoretical Computer Science*, (0):–, 2014.