



# 1 Model Revision of Logical Regulatory Networks 2 using Logic-based Tools


3 **Filipe Gouveia**<sup>1</sup>

4 INESC-ID/Instituto Superior Técnico, Universidade de Lisboa  
5 Rua Alves Redol 9, 1000-029, Lisboa, Portugal  
6 filipe.gouveia@tecnico.ulisboa.pt  
7  <https://orcid.org/0000-0003-1852-2782>

8 **Inês Lynce**<sup>2</sup>

9 INESC-ID/Instituto Superior Técnico, Universidade de Lisboa  
10 Rua Alves Redol 9, 1000-029, Lisboa, Portugal  
11 ines.lynce@tecnico.ulisboa.pt  
12  <https://orcid.org/0000-0003-4868-415X>

13 **Pedro T. Monteiro**<sup>3</sup>

14 INESC-ID/Instituto Superior Técnico, Universidade de Lisboa  
15 Rua Alves Redol 9, 1000-029, Lisboa, Portugal  
16 pedro.tiago.monteiro@tecnico.ulisboa.pt  
17  <https://orcid.org/0000-0002-7934-5495>

## 18 — Abstract —

19 Recently, biological data has been increasingly produced calling for the existence of computational  
20 models able to organize and computationally reproduce existing observations. In particular,  
21 biological regulatory networks have been modeled relying on the Sign Consistency Model or the  
22 logical formalism. However, their construction still completely relies on a domain expert to choose  
23 the best functions for every network component. Due to the number of possible functions for  
24  $k$  arguments, this is typically a process prone to error. Here, we propose to assist the modeler  
25 using logic-based tools to verify the model, identifying crucial network components responsible  
26 for model inconsistency. We intend to obtain a model building procedure capable of providing  
27 the modeler with repaired models satisfying a set of pre-defined criteria, therefore minimizing  
28 possible modeling errors.

29 **2012 ACM Subject Classification** Logic programming and answer set programming

30 **Keywords and phrases** Logical Regulatory Networks, Model Revision, Answer Set Programming,  
31 Boolean Satisfiability, Logic-based tools

32 **Digital Object Identifier** 10.4230/OASICS.ICLP.2018.23

## 33 **1** Introduction

34 Modeling biological regulatory networks is particularly useful to test hypotheses and to  
35 identify predictions *in silico*. With this aim, different qualitative formalisms have been  
36 introduced to model, analyze and simulate regulatory networks and their behaviors. However,  
37 the simulation and analysis of such behaviors is hindered by the combinatorial explosion

<sup>1</sup> [Fundação para a Ciência e a Tecnologia (FCT) PhD grant SFRH/BD/130253/2017]

<sup>2</sup> [National funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013]

<sup>3</sup> [Fundação para a Ciência e a Tecnologia (FCT) project grant PTDC/EEI-CTP/2914/2014]



© Filipe Gouveia, Inês Lynce and Pedro T. Monteiro;  
licensed under Creative Commons License CC-BY

Technical Communications of the 34th International Conference on Logic Programming (ICLP 2018).

Editors: Alessandro Dal Palu', Paul Tarau, Neda Saeedloei, and Paul Fodor; Article No. 23; pp. 23:1–23:10

OpenAccess Series in Informatics



OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

of the qualitative state space. To tackle this problem, formal verification techniques have been introduced in Systems Biology. These techniques include model-checking techniques to automatically verify reachability properties [14], model reduction techniques to reduce the size of the generated dynamics [15], SAT-based approaches to identify attractors [4], among others [17].

Given a complete model of a regulatory network, newly acquired experimental data may render it inconsistent, forcing the model to be revised and updated. The process of review and update a model is called model revision, which is still mainly a manual task performed by a modeler, typically an expert in the domain, and therefore prone to error.

Approaches to model revision relying on the Sign Consistency Model (SCM) have been implemented using logic-based tools such as Answer Set Programming (ASP)[8] and Boolean Satisfiability (SAT)[10]. However, the SCM lacks in expressiveness for regulatory functions, as it is based in sign algebra. This work aims to extend current approaches for model revision to the Logical formalism, and to provide a semi-automatic tool to assist the modeler throughout the model definition process [21].

An overview of some of the key concepts of regulatory networks is given in Section 2. In Section 3 it is mentioned some of the work done in System Biology, regarding regulatory networks. Section 4 describes the logic-based approach for Model Revision. Section 5 concludes the document with an overview of the directions of the future work.

## 2 Regulatory Networks

A biological regulatory network is a set of proteins and genes, that interact with each other or with other substances in the cell. Qualitative models have proven to be well adapted for the modeling of systems where quantitative information is generally incomplete or noisy. Typically, network components only affect other components above some concentration level. In this way, it is possible to consider discrete variables to model regulatory networks, corresponding to different levels of concentration, e.g. active/inactive.

### 2.1 Logical Model

Logical models were used to represent regulatory networks by Kauffman in 1969 [12], and Thomas in 1973 [20].

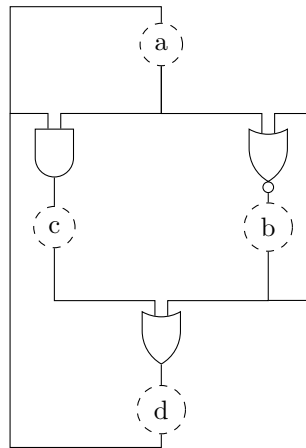
In the Logical Model the components of the network are represented by Boolean variables. A Boolean variable can either be *True* (1, on, active) or *False* (0, off, inactive). If a component in a regulatory network is represented by a Boolean variable, then it has value *True* if it is present (or activated), and it has value *False* if it is absent (or inhibited).

Moreover, the interactions between components are described as Boolean functions [20]. This will allow to determine the state of a component based on the presence or absence of other components.

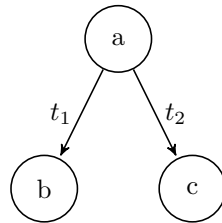
A Logical Model can be represented with a logical circuit since nodes have a Boolean value and regulatory functions are Boolean functions, as shown in Figure 1.

Figure 1 illustrates an example of a logical model with the correspondent regulatory functions. With this representation we can verify that, for example, component  $c$  is regulated by components  $d$  and  $a$ , and its regulatory function is a logical AND from these two inputs.

The (Boolean) Logical Model can be generalized [21]. It is possible to consider more than two values for each variable. For example, considering Figure 2, we can have a variable  $a$  that affects  $b$  above a concentration level threshold  $t_1$ , but only affects  $c$  above a concentration level threshold  $t_2 > t_1$ . In this case variable  $a$  can have three possible values:



■ **Figure 1** Example of a Logical Model represented as a logical circuit.



■ **Figure 2** Example of a Generalization of the Logical Model.

- 83 ■ 0: concentration level below  $t_1$  (not affecting any other variable);
- 84 ■ 1: concentration level between  $t_1$  and  $t_2$  (only affecting variable  $b$ );
- 85 ■ 2: concentration level above  $t_2$  (affecting variables  $b$  and  $c$ );

86 Formally, we can define a Logical Model as a tuple  $(G, K)$  where:

- 87 ■  $G = \{g_1, g_2, \dots, g_n\}$  is the set of components of the network. Each  $g_i$  is associated with
- 88 an integer value in  $\{0, \dots, max_i\}$ , representing the concentration level of the component.
- 89 The state of the network is thus defined as a vector  $s \in S = \prod_{g_i \in G} \{0, \dots, max_i\}$ .
- 90 ■  $K = \{K_1, K_2, \dots, K_n\}$  is the set of regulatory functions where  $K_i$  is the regulatory function
- 91 of  $g_i$  and  $K_i : S \rightarrow \{0, \dots, max_i\}$ .

92 If all  $max_i = 1$ , then we have a Boolean Logical Model, since each  $g_i \in \{0, 1\}$ .

## 93 2.2 Probabilistic Boolean Networks

94 In a logical model, each component regulated by  $k$  other components can have  $2^{2^k}$  possible  
 95 regulatory Boolean functions. Additionally, in some cases experimental data is insufficient  
 96 or there is incomplete knowledge to choose a single regulatory function, where several  
 97 candidates are possible. In other words, possibly several regulatory functions could explain  
 98 the experimental data. With this in mind, the logical model was extended in order to account  
 99 for the uncertainty of the regulatory functions [18].

100 In a Probabilistic Boolean Network (PBN), each component has several regulatory  
 101 functions, each with a given probability associated. These probabilities are determined based  
 102 on the data available, such that it is compatible with prior knowledge of the network. Then,

103 at each time step, and for each component, a regulatory function is selected according to the  
 104 correspondent probabilities, in order to determine its target value.

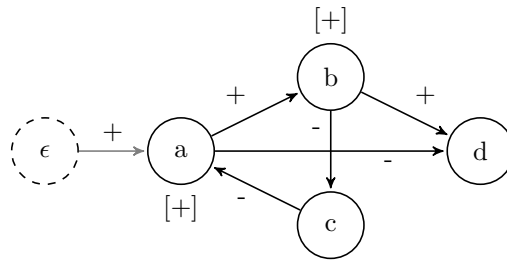
## 105 2.3 Sign Consistency Model

106 Siegel *et al.* proposed a Sign Consistency Model (SCM) [19]. In this approach, it is only  
 107 considered the difference in the expression levels between two situations: a value increase, or  
 108 decrease.

109 The SCM is usually represented by a graph where each node represents a biological  
 110 component, with a value  $+$  (increase of concentration) or  $-$  (decrease of concentration). The  
 111 edges in the graph represent interactions between components and can be labeled “ $+$ ” or  
 112 “ $-$ ”. An edge with label “ $+$ ” (“ $-$ ”) from  $a$  to  $b$  means that an increase of the concentration  
 113 of  $a$  increases (decreases) the concentration of  $b$ .

114 Also, a component can be considered an *input*, having a stimulation from the exterior  
 115 world (outside the regulatory network). If a node is an *input* then its regulatory function  
 116 can be ignored, as there is an exterior stimulation increasing its concentration. In some  
 117 representations, an extra generic node  $\epsilon$  is added to represent the exterior world. For each  
 118 *input* node, an edge is added from  $\epsilon$  to that node.

119 The regulatory functions are then based on the sign algebra, where the value of each com-  
 120 ponent is the sum of the products between the value of each regulator and the corresponding  
 121 edge.



■ **Figure 3** Example of a Sign Consistency Model. Observed components  $a$  and  $b$  are labeled with the correspondent observation. Component  $\epsilon$  represents an external stimulus.

122 Figure 3 illustrates an example of a Sign Consistency Model of a network, where node  $a$   
 123 is an input, and therefore have an input edge from the generic node  $\epsilon$  that represents the  
 124 exterior world. Nodes  $a$  and  $b$  are observed nodes, where an increase of concentration was  
 125 observed. In this example, node  $c$  is expected to have a negative ( $-$ ) sign because it only  
 126 has one regulator  $b$ , which has a negative interaction with  $c$  ( $c = b \times (b \rightarrow c) = (+) \times (-) =$   
 127  $-$ ). However in this example, node  $d$  receives a positive and a negative interaction. in this  
 128 case we say that we have a competition and  $d$  can assume either value.

## 129 3 Related Work

130 The analysis and verification of biological regulatory networks provide opportunities for  
 131 the application of several methodologies. From network identification and parametrization,  
 132 model verification, attractors determination or to model revision. In this section, some of  
 133 the main methods from the last decade are described, as well as the corresponding problem  
 134 and technology.

### 135 3.1 Network and Model Inference

136 Building computational models to correctly represent regulatory networks is of great import-  
137 ance. In order to build such model, one first needs to infer the network topology from a  
138 given set of experimental data. Some of the difficulties of this task relies on the few samples  
139 of observational data and in the incompleteness and inaccuracy of the experimental data.  
140 Then, one also needs to infer, for each component, the associated regulatory functions (model  
141 inference).

142 In regulatory networks inference, several statistical learning techniques are commonly  
143 used [2, 7]. Also, logic-based tools have been successfully used to learn biological models.  
144 *Caspo*[11] is a tool to identify the complete family of feasible models from a training Boolean  
145 logical model from prior knowledge and experimental data.

### 146 3.2 Reachability Verification

147 Given a model and a set of experimental data, it is interesting to verify if the model can  
148 explain the results obtained in the experiment. In particular, one may verify if the model is  
149 capable of generating behaviors from a set of initial states to a set of target states. These  
150 behaviors are typically represented by a State Transition Graph (STG), where nodes represent  
151 states of the network, and edges represent possible transitions between states. The generation  
152 of this STG can be made synchronously or asynchronously. In the synchronous approach, in  
153 a given state of the STG, all components can update their value simultaneously, i.e., each  
154 state as a single successor. In the asynchronous approach, in a given state of the STG, only  
155 one component can update their value to a successor state, i.e., each state has as many  
156 successors as components changing their values.

157 Model checking consists in the verification if a model satisfies a given (set of) property [3],  
158 and has been successfully used for the verification of regulatory networks. Here, biological  
159 observations are encoded in temporal logic formulas, and a model checker is used to verify  
160 the existence of particular behaviors [14].

161 Also of interest, is to know how can a system be influenced in order to avoid reaching unsafe  
162 or undesired states. Recently, the work in [6] introduces the notion of *bifurcation*, transitions  
163 after which a given goal is no longer reachable. This work presents a method using Answer  
164 Set Programming, to identify bifurcations given a model represented as a discrete finite-state  
165 of interacting components. However, since this method relies on under/over approximations,  
166 is not complete, i.e., does not guarantee the identification of all the bifurcations.

### 167 3.3 Attractors Identification

168 A key property of the dynamics of a regulatory network are *attractors*, which typically denote  
169 subsets of states of biological interest. There are two types of attractors: point attractors  
170 and cycle attractors. A point attractor, or a stable state, is a state from which there is no  
171 transition to any other state in the STG. A cycle attractor is a set of states, whose sequence  
172 repeats over time, from which no transition can leave, i.e., a terminal strongly connected  
173 component in the STG.

174 An efficient approach to determine point attractors in (multivalued) logical models  
175 uses Multi-values Decision Diagrams (MDDs) [16]. Also, some approaches consider the  
176 identification of point and cycle attractors in synchronous dynamics. The work in [5] uses  
177 Answer Set Programming (ASP) and allows the determination of all attractors considering  
178 a Markovian program in order to overcome the challenge of determining the number of  
179 time-steps needed to achieve an attractor. The work in [4] uses a SAT based bounded model

180 checker to determine all the attractors of the network by incrementally determining the  
181 attractors of a given length.

### 182 **3.4 Reduction**

183 It is often the case where the generation of the network dynamics is intractable for large  
184 and complex regulatory networks, due to the state space combinatorial explosion. Reduction  
185 techniques can then be applied in order to reduce the model, and therefore the generated  
186 state space. It has been shown that reduction methods can be successfully applied preserving  
187 some dynamical properties of the network, such as attractors [15].

## 188 **4 Model Revision Approach**

189 During the iterative model construction procedure, as new data is acquired, the current model  
190 may not be able to explain the new data, and therefore need to be revised. Revision processes  
191 capable of suggesting addition/removal of networks interactions, and changes to variable  
192 values in order to make a model consistent with the available data have been proposed  
193 [13]. An approach was proposed considering the SCM and developed using the Answer Set  
194 Programming (ASP) paradigm [8]. Also, an approach was proposed using MaxSAT, a SAT  
195 extension used to solve optimization problems [10]. However, the SCM formalism relies on a  
196 simple rule regarding regulatory functions.

197 The logical formalism [20] has been widely used to model biological networks, and have  
198 been successfully implemented using ASP [9] and SAT [1], allowing to model the regulatory  
199 functions with increased expressiveness w.r.t. the SCM. Model repair usually operates under  
200 a minimal assumption as there can be several ways to make a model consistent. Such  
201 optimization criteria can be regarding the number of atomic repair operations [8, 10] or  
202 considering some properties found in the literature [13]. Nevertheless, existing approaches  
203 typically rely on repair operations that potentially change the topology of the network,  
204 invalidating previous domain knowledge.

205 As mentioned in Section 2, there can be several regulatory functions that can explain the  
206 experimental data. Avoiding changing the topology of the network and change regulatory  
207 functions leads to a minimal impact on the truth table of the variables of the model, and  
208 therefore a smaller impact on the associated dynamics.

209 Our idea is to develop a model revision procedure capable of building a consistent  
210 model iteratively as new data is acquired, relying on the logical formalism. Moreover, it is  
211 desired to avoid changing the topology of the network, and try to explain possible causes of  
212 inconsistencies with regulatory functions.

213 On a first phase of the work, one should be able to verify the consistency of a given model  
214 with a set of experimental data, i.e., if the model can explain the experimental data obtained.  
215 Model checking techniques should be used for this purpose. It is intended to implement  
216 this using different logic based tools, such as ASP, SAT and MaxSAT, in order to make a  
217 comparison with respect to the easiness of representation and computational efficiency.

218 On a second phase, if a model is not consistent with the experimental data, the causes  
219 of such inconsistencies must be identified. This is closely related to the identification of  
220 Minimal Unsatisfiable Subsets (MUSes) in SAT formulas, and therefore SAT-based tools  
221 should be used in the process. As there can be multiple concurrent reasons to explain the  
222 existence of inconsistencies, a biological meaningful measure should be provided in order to  
223 rank the possible explanations to be presented to a modeler.

■ **Listing 1** Example of input

```

edge(c1,c2,0).
edge(c1,c3,1).
edge(c2,c1,1).
edge(c2,c3,1).
edge(c4,c2,1).
edge(c4,c3,0).

functionOr(c1,1).
functionAnd(c1,1,c2).

functionOr(c2,1).
functionAnd(c2,1,c1).
functionAnd(c2,1,c4).

obs_vlabel(c1,1).
obs_vlabel(c2,0).
obs_vlabel(c3,0).
obs_vlabel(c4,0).

functionOr(c3,1..2).
functionAnd(c3,1,c1).
functionAnd(c3,2,c2).
functionAnd(c3,2,c4).

```

224 On a final phase, considering the most plausible cause for inconsistency, a procedure for  
 225 model revision should be defined. For this, SAT-based tools for the identification of Minimal  
 226 Correction Subsets (MCSes) should be considered. This model revision process should be  
 227 iterative, considering that multiple reasons for inconsistency may exist. We will first try  
 228 to explain the causes of inconsistencies with regulatory functions. However, changing the  
 229 regulatory functions may not be sufficient, and therefore one may need to consider changing  
 230 the topology of the network. To achieve this, an iterative approach will be considered where  
 231 different causes of inconsistency are taken into account.

232 In the revision process, not only the model consistency must be taken into account, but  
 233 also other known properties about the network must hold, such as the existence of known  
 234 attractors and its reachability. As the number of possible states for a network increases  
 235 exponentially with the number of components, guaranteeing the existence of the known  
 236 attractors, for example, can be a difficult task. For this, model reduction techniques may be  
 237 necessary.

238 We start by considering only monotone non-degenerate functions. In a monotone function,  
 239 each regulator has only one role, i.e., it is either strictly positive or negative. In a non-  
 240 degenerate function, all regulators are functional, i.e., all regulators have an influence in the  
 241 regulatory function.

242 Currently, we have an Answer Set Programming approach implemented for the logical  
 243 formalism, and we are able to verify the consistency of a model given some experimental  
 244 data at steady state, i.e., without considering any dynamics. Moreover, we are able, in case  
 245 of inconsistency, to identify the regulatory functions that can explain such inconsistencies.  
 246 We are working on the process of repairing such functions in order to validate if the proposed  
 247 model solutions become consistent.

248 We represent the logical model as a directed graph and the regulatory functions in  
 249 disjunctive normal form (DNF). As we only consider monotone functions and, therefore, each  
 250 regulator only has one role (positive interaction or negative interaction), this role is defined  
 251 by the edge. A positive (negative) edge represents a positive (negative) interaction. An  
 252 example is presented in Listing 1 with the representation of the model and the observations.

253 The predicate `edge(A,B,S)` represents an edge from `A` to `B` with sign `S`. Predicate  
 254 `functionOr(A,C)` indicates the number of clauses (`C`) in the regulatory function of `A` in the  
 255 DNF. Predicate `functionAnd(A,C,B)` indicates that the clause `C` of the regulatory function  
 256 of `A` contains variable `B`. The observations are represented by the predicate `obs_vlabel(A,S)`,



■ **Listing 2** Consistency check in Answer Set Programming

```

sign(0;1).                                complement(T,S):-sign(S),sign(T),T!=S.

vertex(V):-edge(V,_,_).                    vertex(V):-edge(_,V,_).

% generate
1{vlabel(V,S):sign(S)}1:-vertex(V).
{r_gen(V)} :- vertex(V).                  {r_part(V)} :- vertex(V).

:-vlabel(V,S), obs_vlabel(V,T),complement(S,T).

% functions
% one positive or negative contribution in a clause
onePositive(V,Id):-functionAnd(V,Id,V2),edge(V2,V,S),vlabel(V2,S).
oneNegative(V,Id):-functionAnd(V,Id,V2),edge(V2,V,S),vlabel(V2,T),
                    complement(S,T).

% none negative contribution in a clause
noneNegative(V,Id):-onePositive(V,Id),not oneNegative(V,Id).

vlabel(V,1):-1{noneNegative(V,Id):functionOr(V,Id)},vertex(V),
              not r_part(V).
vlabel(V,0):-{noneNegative(V,Id):functionOr(V,Id)}0,vertex(V),
              not r_gen(V).

repair(f,V) :- r_gen(V).                   repair(f,V) :- r_part(V).
#minimize {1,V : repair(_,V)}.

```

257 which means that value S was observed in node A.

258 The main idea behind the encoding presented in Listing 2 is that each node of the network  
259 (**vertex**) must have exactly one label that represents the expected value (**vlabel**), and it  
260 is not possible to have a label different from the observation. Each label is determined  
261 based on the contributions of each regulator in the associated regulatory function. To allow  
262 determining possible causes of inconsistencies, we defined the predicates **r\_gen** and **r\_part**  
263 indicating that a regulatory function should be generalized or particularized, respectively,  
264 justifying the inconsistency of the model. In order to achieve this, we allow a label of a vertex  
265 to be different than expected given the regulatory function, if that function is a possible  
266 cause of inconsistency.

## 267 5 Conclusions and Future Work

268 Qualitative formalisms have been used whenever information is scarce. In particular, the  
269 logical formalism has proved successful to model complex biological networks. Nevertheless,  
270 the construction of such models is still mainly a manual task, and therefore prone to errors  
271 and to interpretations of a specific modeler. Here, we focus on the problem of model revision,  
272 i.e., to assist the modeler in the process of revising the model associated functions in order  
273 to render the model consistent with the existing and new data.

274 Here, we propose to consider the logical formalism limiting to the set of monotone non-  
275 degenerate functions. Also, we start by verifying the consistency of models at steady state,  
276 i.e., without considering any dynamics. We consider an Answer Set Programming approach



277 to identify which nodes are the causes for model inconsistency.

278 We intend to follow the work plan described in the previous section, and be able to  
 279 present a procedure and corresponding tool capable of building a consistent model iteratively  
 280 as new data is acquired.

## 281 ——— References ———

- 282 **1** Armin Biere, Marijn Heule, and Hans van Maaren. *Handbook of satisfiability*, volume 185.  
 283 IOS press, 2009.
- 284 **2** Atul J Butte and Isaac S Kohane. Mutual information relevance networks: functional  
 285 genomic clustering using pairwise entropy measurements. In *Biocomputing 2000*, pages  
 286 418–429. World Scientific, 1999.
- 287 **3** Edmund M Clarke, Orna Grumberg, and Doron Peled. *Model checking*. MIT press, 1999.
- 288 **4** Elena Dubrova and Maxim Teslenko. A sat-based algorithm for finding attractors in syn-  
 289 chronous boolean networks. *IEEE/ACM transactions on computational biology and bioin-*  
 290 *formatics*, 8(5):1393–1399, 2011.
- 291 **5** Timur Fayruzov, Jeroen Janssen, Dirk Vermeir, Chris Cornelis, and Martine De Cock. Mod-  
 292 elling gene and protein regulatory networks with answer set programming. *International*  
 293 *journal of data mining and bioinformatics*, 5(2):209–229, 2011.
- 294 **6** Louis Fippo Fitime, Olivier Roux, Carito Guziolowski, and Loïc Paulevé. Identification  
 295 of bifurcation transitions in biological regulatory networks using answer-set programming.  
 296 *Algorithms for Molecular Biology*, 12(1):19, 2017.
- 297 **7** Nir Friedman. Inferring cellular networks using probabilistic graphical models. *Science*,  
 298 303(5659):799–805, 2004.
- 299 **8** Martin Gebser, Carito Guziolowski, Mihail Ivanchev, Torsten Schaub, Anne Siegel, Sven  
 300 Thiele, and Philippe Veber. Repair and prediction (under inconsistency) in large biological  
 301 networks with answer set programming. In *KR*, 2010.
- 302 **9** Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. Answer set  
 303 solving in practice. *Synthesis Lectures on Artificial Intelligence and Machine Learning*,  
 304 6(3):1–238, 2012.
- 305 **10** João Guerra and Inês Lynce. Reasoning over biological networks using maximum satis-  
 306 fiability. In *Principles and Practice of Constraint Programming*, pages 941–956. Springer,  
 307 2012.
- 308 **11** Carito Guziolowski, Santiago Videla, Federica Eduati, Sven Thiele, Thomas Cokelaer, Anne  
 309 Siegel, and Julio Saez-Rodriguez. Exhaustively characterizing feasible logic models of a  
 310 signaling network using answer set programming. *Bioinformatics*, page btt393, 2013.
- 311 **12** Stuart Kauffman. Homeostasis and differentiation in random genetic control networks.  
 312 *Nature*, 224(5215):177, 1969.
- 313 **13** Elie Merhej, Steven Schockaert, and Martine De Cock. Repairing inconsistent answer set  
 314 programs using rules of thumb: A gene regulatory networks case study. *International*  
 315 *Journal of Approximate Reasoning*, 83:243–264, 2017.
- 316 **14** Pedro T Monteiro, Wassim Abou-Jaoudé, Denis Thieffry, and Claudine Chaouiya. Model  
 317 checking logical regulatory networks. *IFAC Proceedings Volumes*, 47(2):170–175, 2014.
- 318 **15** Aurélien Naldi, Elisabeth Remy, Denis Thieffry, and Claudine Chaouiya. Dynamically con-  
 319 sistent reduction of logical regulatory graphs. *Theoretical Computer Science*, 412(21):2207–  
 320 2218, 2011.
- 321 **16** Aurélien Naldi, Denis Thieffry, and Claudine Chaouiya. Decision diagrams for the repres-  
 322 entation and analysis of logical models of genetic networks. In *CMSB*, volume 7, pages  
 323 233–247. Springer, 2007.

## 23:10 Logic-based approach for Model Revision

- 324 **17** Loïc Paulevé. Reduction of qualitative models of biological networks for transient dynamics  
325 analysis. *IEEE/ACM transactions on computational biology and bioinformatics*, 2017.
- 326 **18** Ilya Shmulevich, Edward R Dougherty, Seungchan Kim, and Wei Zhang. Probabilistic  
327 boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioin-*  
328 *formatics*, 18(2):261–274, 2002.
- 329 **19** Anne Siegel, Ovidiu Radulescu, Michel Le Borgne, Philippe Veber, Julien Ouy, and  
330 Sandrine Lagarrigue. Qualitative analysis of the relation between dna microarray data  
331 and behavioral models of regulation networks. *Biosystems*, 84(2):153–174, 2006.
- 332 **20** René Thomas. Boolean formalization of genetic control circuits. *Journal of theoretical*  
333 *biology*, 42(3):563–585, 1973.
- 334 **21** René Thomas. Regulatory networks seen as asynchronous automata: a logical description.  
335 *Journal of theoretical biology*, 153(1):1–23, 1991.