HAL
open science

Boolean Network Identification from Perturbation Time Series Data combining Dynamics Abstraction and Logic Programming

Max Ostrowski, Loïc Paulevé, Torsten Schaub, Anne Siegel, Carito Guziolowski

To cite this version:

Max Ostrowski, Loïc Paulevé, Torsten Schaub, Anne Siegel, Carito Guziolowski. Boolean Network Identification from Perturbation Time Series Data combining Dynamics Abstraction and Logic Programming. BioSystems, 2016, 10.1016/j.biosystems.2016.07.009 . hal-01354075

HAL Id: hal-01354075

https://hal.science/hal-01354075

Submitted on 17 Aug 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Boolean Network Identification from Perturbation Time Series Data combining Dynamics Abstraction and Logic Programming

M. Ostrowski[a,1], L. Paulevé[c,1], T. Schaub[a,b], A. Siegel[d], C. Guziolowski[e,*]

[a] *University of Potsdam, Potsdam, Germany*
[b] *INRIA, Rennes, France*
[c] *CNRS, Université Paris-Sud LRI-UMR 8623, Orsay, France*
[d] *CNRS, Université de Rennes 1, IRISA-UMR 6074, Rennes, France*
[e] *École Centrale de Nantes, IRCCyN UMR CNRS 6597, Nantes, France*

Abstract

Boolean networks (and more general logic models) are useful frameworks to study signal transduction across multiple pathways. Logic models can be learned from a prior knowledge network structure and multiplex phosphoproteomics data. However, most efficient and scalable training methods focus on the comparison of two time-points and assume that the system has reached an early steady state. In this paper, we generalize such a learning procedure to take into account the time series traces of phosphoproteomics data in order to discriminate Boolean networks according to their transient dynamics. To that end, we identify a necessary condition that must be satisfied by the dynamics of a Boolean network to be consistent with a discretized time series trace. Based on this condition, we use Answer Set Programming to compute an over-approximation of the set of Boolean networks which fit best with experimental data and provide the corresponding encodings. Combined with model-checking approaches, we end up with a global learning algorithm. Our approach is able to learn logic models with a true positive rate higher than 78% in two case studies of mammalian signaling networks; for a larger case study, our method provides optimal answers after 7 minutes of computation. We quantified the gain in our method predictions precision compared to learning approaches based on static data. Finally, as an application, our method proposes erroneous time-points in the time series data with respect to the optimal learned logic models.

Keywords: Model identification, Time series data, Multiplex phosphoproteomics data, Boolean networks, Answer Set Programming

1. Introduction

Generic prior knowledge about canonical cell signaling networks can be retrieved from database sources. This provides a first insight on how cells respond to their environment by triggering processes such as growth, survival, apoptosis (cell death), and migration. However, little is known about the exact chaining and composition of signaling events within these networks in specific cells and in response to specific experimental perturbations, as provided by the simulations of predictive mathematical models, e.g. a set of differential equations or a set of logic rules. When building predictive models, the parameters of a model (built according to generic prior knowledge) can be fitted to the data to obtain the most plausible model for a specific cell type, if enough experimental data is available. This is normally achieved by defining an objective fitness function to be optimized. In this context, post-translational modifications, notably protein phosphorylation, play a key role in signaling. They are very useful for the training of model parameters through the use of multiplex

*Corresponding author
Email address: `Carito.Guziolowski@irccyn.ec-nantes.fr` (C. Guziolowski)
[1] Co-first authors

phosphorylation assays, a recent form of high-throughput data providing information about protein-activity modifications in a specific cell type upon various experimental perturbations (clamping) [1].

Boolean logical networks [2] provide a simple yet powerful qualitative framework which has become very popular during the last decade to model signaling or regulatory networks [3]. In contrast to quantitative methods which permit fine-grained kinetic analysis, qualitative approaches allow for addressing large-scale biological networks. In this context, the manual identification of logic rules underlying the system has been addressed under different hypotheses and methods [4]. In particular, scalable methods restrain themselves to learning models from two time points (start; end) and assume the system has reached an early steady-state when the measurements are performed. As shown in [5], this assumption prevents capturing important characteristics of signaling networks such as feedback cycles.

Data-driven methods for learning causal graphs representing molecular networks have been widely proposed [6, 7, 8]. We here focus on methods learning causal graphs from time series data. Some of such methods use ODE's kinetic modeling to build dynamic predictive models from time series gene expression data [9, 10]. Recently, a crowdsourcing challenge was proposed to learn causal graphs for breast cancer cell lines using multiplex phosphoproteomic time series data. The results in [11] show that methods based on machine learning techniques, using only data and no prior knowledge network, obtained a significant score. This study reported as well that methods using prior-knowledge outperformed methods solely based on data. Having said that, the task of evaluation of such methods is very delicate since an exhaustive experimental verification of a small-scale causal graph is not feasible. Furthermore, other types of evaluation such as verifying model predictions, do not necessarily require a causal graph model structure. The approach we propose here belongs to the category of methods that use time series phosphoproteomics data and a prior-knowledge graph to infer logical causal models that can be simulated *a-posteriori* using synchronous or asynchronous updates.

Our method infers Boolean networks (BNs) from time series datasets and it scales to the size of currently studied BNs. Given multiplex time series data from the measurement of a partial set of biological entities under different experimental perturbations, we want to identify all the BNs that have a structure compatible with a given prior knowledge interaction graph and that can reproduce all the (experimentally) observed time series. Time series data are assumed incomplete, i.e., only a subset of network components are observed, with measurements made at discrete time points and with normalized continuous values. It is possible that no BN, constrained by the prior interaction graph, reproduces all the input time series. In such a case, we introduce a fitness function to measure the distance between a trace of a BN simulation and a measured time series. Therefore, we aim to infer the BNs whose dynamics contains traces with the best fitness to all measurements.

Our approach relies on the combination of several techniques. First, we introduce a necessary condition for discretized time series data to be the trace of a BN. This provides an over-approximation of the successive reachability properties, allowing to reject BNs which cannot reproduce the time series without a costly exhaustive analysis of the dynamics. Then, we use Answer Set Programming (ASP) to enumerate BNs which approximate the best experimental data while satisfying the necessary condition on the dynamics. As a result, we obtain a set of BNs associated with traces which both satisfy the necessary condition and optimally fit with experimental data. Because of the over-approximation of reachability, a part of the returned BNs cannot reproduce the associated Boolean traces. Such false positives can be detected *a posteriori* using model-checking on the returned results.

This paper extends the results presented in [12] in several ways: a complete and detailed characterization of the method, illustrated step by step with a toy example; a detailed description of the ASP implementation of the inference and optimization, together with justifications for the computation of the fitness of predictions and experimental observations; and with a general benchmark, composed of three case studies, which increases the diversity of networks against which we evaluate our method.

In order to evaluate our method we used synthetic data generated from BNs of three mammalian signaling networks induced by the: (i) Epidermal Growth Factor (EGF) and Tumor Necrosis Factor alpha (TNFα), (ii) T-cell Receptor (TCR), and (iii) Epidermal Growth Factor Receptor (ERBB). These networks have between 13-40 nodes and 16-50 edges and contain multiple cycles. Our prototype implementation was able to identify efficiently all BNs satisfying the necessary condition with a rate of true positives over 78% for

networks of less than 25 edges. We measured the impact of incomplete networks on the precision of learned BNs; and estimated the added-value of models identified with our method on the full time series with respect to models learned from two time points, considered as a steady state. Finally we present an application of this method to detect erroneous time-points in the time series data with respect to the learned BNs.

2. Boolean Network Identification

2.1. Admissible Boolean Networks and multiplex time series data

Boolean Networks (BNs). A BN with n components $\{1, \ldots, n\}$ consists of a tuple of n Boolean functions $F = (f_1, \ldots, f_n)$ where each function $f_i : \mathbb{B}^n \to \mathbb{B}$, $\mathbb{B} \triangleq \{0, 1\}$, $i \in \{1, \ldots, n\}$, associates with each global state $x \in \mathbb{B}^n$ of the network the next value of the i-th component. The value of the i-th component in x is denoted x_i.

As a toy example that is used all along the present article, let us consider the BN depicted in Fig. 1b with four components nI, nJ, nA, and nB in $\mathbb{B}$ associated to the following boolean functions:

$$F : \begin{cases} f_{nI}(nI, nJ, nA, nB) & = & 0 \\ f_{nJ}(nI, nJ, nA, nB) & = & \neg\, nI \\ f_{nA}(nI, nJ, nA, nB) & = & nI \wedge \neg\, nB \\ f_{nB}(nI, nJ, nA, nB) & = & nJ \vee nA \end{cases}$$

Transition relation and associated semantics. The transitions between global states of the network are specified with a transition relation $\to \subseteq \mathbb{B}^n \times \mathbb{B}^n$. The transitive closure of $\to$ is denoted by $\to^*$. Given $x, x' \in \mathbb{B}^n$, $x \to^* x'$ if and only if, either $x = x'$ or $x \to \cdots \to x'$.

Several definitions of the transition relation $\to$ can be used depending on the update schedule of the components [13], ranging from so-called parallel (or synchronous) updates where each transition updates the value of all the components, to the asynchronous update where each transition updates the value of only one component chosen non-deterministically.

As the over-approximation results presented in this article are independent of the update schedule, we use the general definition, where any number of components can be updated during a transition: for any $x, x' \in \mathbb{B}^n$,

$$x \to x' \overset{\Delta}{\Leftrightarrow} (x \neq x') \wedge (\forall i \in \{1, \ldots, n\}, x'_i \neq x_i \Rightarrow x'_i = f_i(x)) \ . \tag{1}$$

For example, in our toy example, we have that $F(1, 1, 0, 0) = (1, 0, 1, 1)$. This means that three variables may change their value from the state $(1, 1, 0, 0)$. Therefore, we have that $(1, 1, 0, 0) \to (1, 0, 1, 1)$ is a synchronous update scheme, whereas $(1, 1, 0, 0) \to (1, 0, 0, 0)$, $(1, 1, 0, 0) \to (1, 1, 1, 0)$ and $(1, 1, 0, 0) \to (1, 0, 0, 1)$ are valid in an asynchronous update. In our framework, we consider as valid the eight transitions $(1, 1, 0, 0) \to (1, b, c, d)$ where $b, c, d \in \mathbb{B}$.

Prior Knowledge Network and admissible BNs. An *interaction graph* between n components is a digraph between nodes $\{1, \ldots, n\}$ where each edge is signed, i.e., either positive or negative. The interaction graph of a BN F, noted $\mathsf{IG}(F)$, has a positive (resp. negative) edge from node j to node i if and only if there exists $x, x' \in \mathbb{B}^n$ which are identical except on the j-th coordinate where $x_j = 0$ and $x'_j = 1$ and such that $f_i(x) < f_i(x')$ (resp. $f_i(x) > f_i(x')$). Notice that according to this definition, an interaction may contain multiple edges with different signs. The interaction graph of our toy example is depicted in Fig. 1a. Notice that in this graph nI is a specific node since it has no predecessor. We call it an input node.

In the rest of the paper, the *Prior Knowledge Network* (PKN) is an interaction graph which delimits the set of *admissible BNs*: a BN F is admissible with respect to a PKN $\mathcal{G}$ if and only if $\mathsf{IG}(F)$ is a sub-graph of $\mathcal{G}$ and $\mathsf{IG}(F)$ has at most one (signed) edge between two nodes.

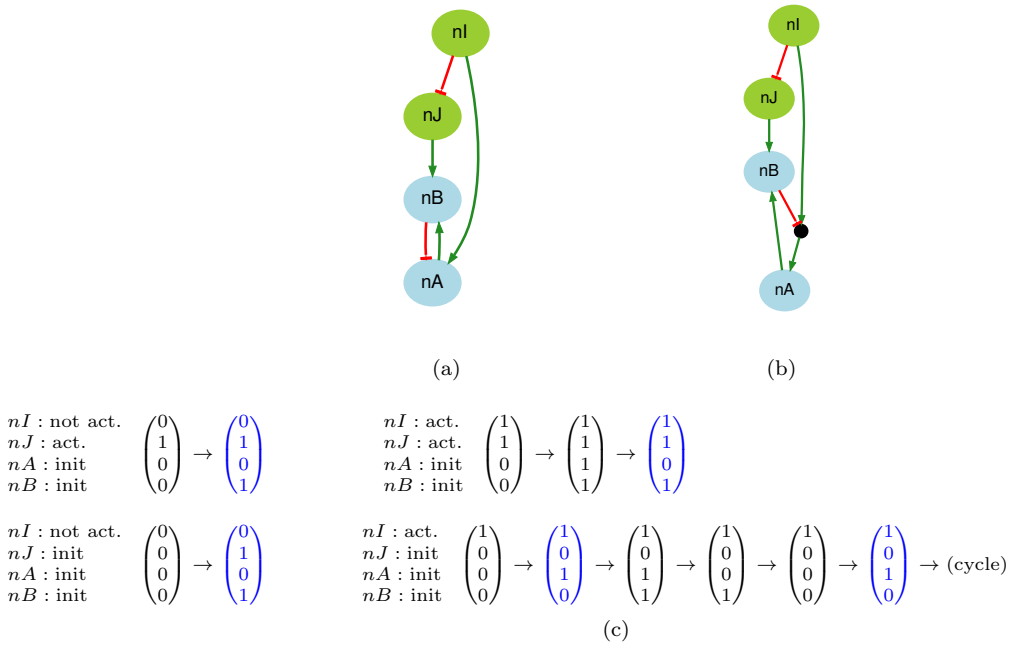


Figure 1: (a) PKN used for the toy example with its associated logical network (b). In (b), the *AND* gate in $f_{nA} = \neg nB \wedge nI$ is represented by a black circle whereas in (a), the *OR* gate in $f_{nA} = \neg nB \vee nI$ is represented by two distinct edges. (c) Four synchronous trajectories associated to the toy example for a boolean network where nI and nJ are candidates to be activated by force and the initial state (*init*) is set to $(0, 0, 0, 0)$; *act.* stands for activated. Blue states highlight the evolution of the state transition, either cyclic or stable behavior.

Biological experimental setting. We consider classical biology experimental settings where the activity of a subset of biological species is observed over time at discrete time points in response to different experimental perturbations. These perturbations consist of a choice of species whose level can be fixed by force and that are modeled by *clamping* operations. Given a BN $F = (f_1, \ldots, f_n)$, assuming that an experimental perturbation consists of a subset A of components with a forced activation, and a subset I of components with a forced inhibition, the corresponding *clamped BN* $F_{[A,I]} = (f'_1, \ldots, f'_n)$ is defined for all $i \in \{1, \ldots, n\}$ as:

$$f'_i \triangleq \begin{cases} x \mapsto 1 & \text{if } i \in A \\ x \mapsto 0 & \text{if } i \in I \\ f_i & \text{otherwise.} \end{cases}$$

In our toy example, we assume that both nI and nJ are candidates for a forced activation. Therefore, there are four different possible configurations of activations which all correspond to a different experimental perturbation. In Fig. 1c, we show the synchronous trajectories of the network upon the four configurations.

BN associated to phosphoproteomic Time Series Data. The analysis of multiplex phosphoproteomics time series data is based on a BN framework formalism. We assume that we are given a PKN describing interactions between compounds involved in a biological signaling network. Some nodes of the PKN are preliminary defined to be either stimuli or inhibitors. In most cases-studies using multiplex phosphoproteomics time series data, all nodes of the PKN with no predecessors are assumed to be *stimuli*. These nodes will define the biological system boundary and correspond to points where an experimentalist can interact with the system in order to stimulate it. Usually *stimuli* represent ligand molecules which bind cellular receptors. They are activated in the system before experimental measurements and will stay unchanged across all time

points during the experimentation. The inhibitor nodes correspond to species chosen because they can be blocked by highly selective small-molecule inhibitors. These highly selective inhibitors are usually added into the system 24 hours before measurements. We model inhibitor nodes as inactive nodes that will stay unchanged across all time points of the experimentation. Prior to the activation or the inhibition of nodes, all compounds of the system are assumed to be at their initial state, which is usually measured as a control experiment. Together, the analysis of phosphoproteomics time series data enters in a general scheme where a PKN is associated to: (i) a set of candidates for forced activations, (ii) a set of candidates for forced inhibitions, and (iii) an initial state corresponding to a control experiment. In order to gain in generality, we will not restrict ourselves to the case where candidates for activations are all nodes without predecessors. We also handle the case when input nodes are inhibitors. Therefore, in the following, we will simply assume that candidates for activations or inhibitors are spread along the PKN.

Time series data correspond to the measurement of phosphorylation activity of a subset of compounds, which may (or not) include inhibitors or stimuli. Without loss of generality, we assume that the time series data are related to the observation of $m \leq n$ nodes that match the nodes $\{1, \ldots, m\}$ of the BN (so the nodes $\{m+1, \ldots, n\}$ are not observed). The observations consist of normalized continuous values: a time series of k data points is denoted by $T = (t^1, \ldots, t^k)$, with $\forall j \in \{1, \ldots, k\}, t^j \in [0; 1]^m$.

Hereafter, we consider a simple binarization of observations using a 0.5 threshold: given a continuous observation $t_i^j \in [0; 1]$ of a component, its Boolean value is noted $\eta(t_i^j)$, where $\eta(t_i^j) \triangleq 1$ when $t_i^j \geq 0.5$ and $\eta(t_i^j) \triangleq 0$ otherwise.

2.2. Over-approximation of Boolean network verification

Given a BN F and a pair of states $x, y \in \mathbb{B}^n$, checking the reachability of y from x ($x \rightarrow^* y$) is a standard model-checking task, known to have a limited scalability due to its theoretical complexity (PSPACE-complete [14]). In this section, we introduce a so-called *meta-state semantics ($\rightrightarrows$)* for BNs. From this semantics, we express a necessary condition for reachability in the concrete semantics ($\rightarrow$), referred to as *support consistency* ($\rightsquigarrow^*$). Meta-state semantics offers properties (notably monotony) that make support consistency efficient to verify, in particular with ASP. However, support consistency is not a sufficient condition for reachability, so this approach may lead to false positives but guarantees the absence of false negatives. Therefore, we will apply exact model-checking approaches on the inferred BNs in order to rule out false positives. Thanks to the over-approximation criteria, one can expect that the set of BNs satisfying the necessary condition is small compared to the full domain of BNs delimited by the PKN, leading to a global gain in terms of performance.

Meta-state semantics. A *meta-state* u of dimension n is a vector of n non-empty subsets of $\mathbb{B}$, noted $\mathbb{M} \triangleq \{\{0\}, \{1\}, \{0, 1\}\}$; the set of meta-states is $\mathbb{M}^n$. In the following, meta-states characterize a set of Boolean states: a state $x \in \mathbb{B}^n$ belongs to a meta-state $u \in \mathbb{M}^n$, noted $x \in u$, iff each Boolean component x_i belongs to the set u_i, i.e., $\forall i \in \{1, \ldots, n\}, x_i \in u_i$. Given a state $x \in \mathbb{B}^n$, $\overline{x}$ is the meta-state such that $\forall i \in \{1, \ldots, n\}, \overline{x}_i = \{x_i\}$. In the scope of a BN $F = (f_1, \ldots, f_n)$, we define a transition relation between meta-states $\rightrightarrows \subseteq \mathbb{M}^n \times \mathbb{M}^n$ as follows: from a meta-state u, there is one transition for each $i \in \{1, \ldots, n\}$ which adds to u_i all the possible values of the function f_i applied to every $x \in u$:

$$u \rightrightarrows v \overset{\Delta}{\Leftrightarrow} u \neq v \wedge \exists i \in \{1, \ldots, n\}, v = \langle u_1, \ldots, u_i \cup \{f_i(x) \mid x \in u\}, \ldots u_n \rangle \ . \tag{2}$$

Several properties arise from this definition, in particular $u \rightrightarrows v$ implies that $\forall i \in \{1, \ldots, n\}$, $u_i \subseteq v_i$; therefore $x \in u \Rightarrow x \in v$ (monotony). Moreover, $u_i \neq v_i$ if and only if $v_i = \{0, 1\}$ and if there exists $x \in u$ such that $f_i(x) \notin u_i$. A trace in meta-state semantics is therefore of length at most n, contains no loop, and converges towards a unique fixed point, which depends on the initial state of the trace. Indeed, by Equation (2), the ordering of meta-state transitions has no impact on the possible modifications: from u, if one can modify both the components i and j, the modification of j (resp. i) will still be possible after updating i (resp. j).

In Fig. 2, we show both the transition graph of the toy example limited to the trajectories started from $(1, 0, 0, 0)$ and $(0, 1, 1, 0)$ without clamping of nJ, and the corresponding meta-semantics transition graph.

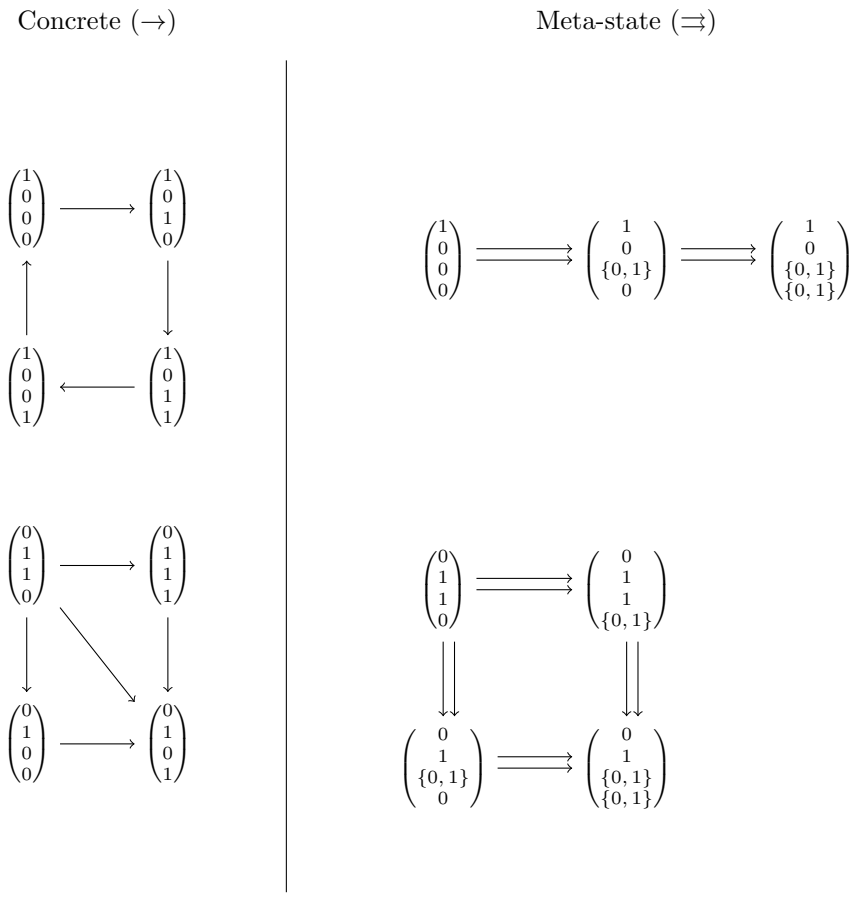


Figure 2: Transition graph of the Boolean network associated to the toy example and transition graph of the meta-semantics. The transitions are limited to nodes accessible from the initial state $(1, 0, 0, 0)$ and $(0, 1, 1, 0)$.

In this figure, we see that the cycle of the BN transition graph is over-approximated by a fixed-point in the meta-state semantics.
Lemma 1 establishes the consistency of the meta-semantics ($\rightrightarrows$) with the concrete semantics ($\rightarrow$): given $x, y \in \mathbb{B}^n$, $x \rightarrow y$ requires that there exists a meta-state u such that $y \in u$ and $\overline{x} \rightrightarrows^* u$, where $\rightrightarrows^*$ is the transitive closure of $\rightrightarrows$.

Lemma 1. $\forall x, y \in \mathbb{B}^n,\ x \rightarrow y \Longrightarrow \exists u \in \mathbb{M}^n, y \in u : \overline{x} \rightrightarrows^* u.$

Proof. Assuming $x \rightarrow y$, let us define the set $I \triangleq \{i \in \{1, \ldots, n\} \mid y_i \neq x_i\}$. From Equation (1), $\forall i \in I, y_i = f_i(x)$. Let us assume that for some strict subset $J \subsetneq I$, $\exists v \in \mathbb{M}^n, \overline{x} \rightrightarrows^* v$ with $\forall i \in J$, $y_i \in v_i$. It is notably the case with $J = \emptyset$. By induction, we show that, for any $k \in I \setminus J$, $\exists u \in \mathbb{M}^n$ such that $\overline{x} \rightrightarrows^* v \rightrightarrows u$ with $\forall i \in J \cup \{k\}, y_i \in u_i$. Remarking that $x \in v$ and defining $u \in \mathbb{M}^n$ such as $u_i = v_k \cup \{f_k(z) \mid z \in v\}$ if $i = k$ and $u_i = v_i$ if $i \neq k$, we obtain that $v \rightrightarrows u$, with $y_k = f_k(x) \in u_k$. □

By induction, Lemma 1 gives a necessary condition for reachability in the concrete semantics: y is reachable from x ($x \rightarrow^* y$) only if there exists a meta-state u such that $y \in u$ and $\overline{x} \rightrightarrows^* u$.

Such a necessary condition for reachability can be furthermore refined by focusing on those components $i \in \{1, \ldots, n\}$ that are equal in x and y ($x_i = y_i$) and such that we have $u_i = \{0, 1\}$ for all meta-states u containing y with $\overline{x} \rightrightarrows^* u$. In this case, the refinement assumption ensures that all such u's also contain a state z with $f_i(z) = y_i = x_i$. Intuitively, this refinement ensures that if the i-th component has to temporarily change its value for reaching y, a state from which it can recover its initial (and final) value has

to be reached in between. Such a condition is referred to as *support consistency* (Definition 1). Theorem 1 states that support consistency is a necessary condition for reachability.

Definition 1 (Support Consistency ($\rightsquigarrow^*$))**.** A state $x \in \mathbb{B}^n$ is *support-consistent* with $y \in \mathbb{B}^n$, denoted by $x \rightsquigarrow^* y$, if and only if there exists $u \in \mathbb{M}^n$ with $\overline{x} \rightrightarrows^* u$ such that $y \in u$ and for all $i \in \{1, \dots, n\}$

- either $y_i \neq x_i$,
- or $y_i = x_i$ and $u_i \neq \{0, 1\}$
- or $y_i = x_i$ and $u_i = \{0, 1\}$, and then there exists $z \in u$ such that $f_i(z) = y_i$.

Theorem 1. *$\forall x, y \in \mathbb{B}^n$, $x \rightarrow^* y \Longrightarrow x \rightsquigarrow^* y$.*

Proof. Let us consider any tuple of states $(x^1, \dots, x^k)$ with $x^1 = x$, $x^k = y$, and $\forall j \in \{1, \dots, k-1\}, x^j \rightarrow x^{j+1}$. From Lemma 1 and the monotony of $\rightrightarrows$, there exists $u \in \mathbb{M}^n$ such that $\overline{x} \rightrightarrows^* u$ with $\forall j \in \{1, \dots, k\}$, $x^j \in u$. If for all such u, for any $i \in \{1, \dots, n\}$, $u_i = \{0, 1\}$ implies that there exists $l \in \{1, \dots, k\}$ with $x^l_i \neq x_i$. If $y_i = x_i$, there necessarily exists $m \in \{l, \dots, k-1\}$ such that $f_i(x^m) = y_i$. Therefore $x^m \in u$. $\square$

2.3. Optimization with respect to time series data

Our objective is to infer BNs that are admissible with a given PKN and that verify the sequential reachability of binary states in $\mathbb{B}^m$ as close as possible to a given time series data and its associated experimental settings.

The distance between a binary sequence $X = (x^1, \dots, x^k)$ having values in $\mathbb{B}$, and a time series $T = (t^1, \dots, t^k)$ having continuous values within $[0; 1]$ is evaluated with the standard *Mean Squared Error*:

$$\mathsf{mse}(X, T) \triangleq \sqrt{\sum_{j=1}^{k} \sum_{i=1}^{m} \left(x_i^j - t_i^j\right)^2} \ .$$

Distance between a time series data and a BN. Given a time series T with associated clamping A, I, the distance between a BN F and (T, A, I), noted $\mathsf{mse}(F_{[A,I]}, T)$, is the minimal MSE between T and a sequence of binary states $X = (x^1, \dots, x^k)$, with $\forall j \in \{1, \dots, k\}, x^j \in \mathbb{B}^n$, which are successively reachable in $F_{[A,I]}$: $x^1 \rightarrow^* x^2 \dots \rightarrow^* x^k$. Remark that that the MSE depends only on the first m dimensions of the states. We notice that the lowest possible $\mathsf{mse}(X, T)$ among all Boolean traces is the MSE between T and its binarization $\eta(T) = ((\eta(t_i^1))_{i=1\dots m}, \cdots, (\eta(t_i^k))_{i=1\dots m})$. Let us call $\mathrm{MSE}_T \triangleq \mathsf{mse}(\eta(T), T)$ this *minimum MSE* which is intrinsic to the time series T and to the threshold for binarization (0.5); notably, $\mathsf{mse}(F_{[A,I]}, T) \geq \mathrm{MSE}_T$. Whenever $\mathsf{mse}(F_{[A,I]}, T) = \mathrm{MSE}_T$, we say the BN F *reproduces* the time series data T.

Relaxing the semantics constraint. In order to prevent an exhaustive exploration of the BN dynamics for characterizing the sequences of reachable ($\rightarrow^*$) Boolean states, we consider any sequence $X = (x^1, \dots, x^k)$, with $\forall j \in \{1, \dots, k\}, x^j \in \mathbb{B}^n$, that are *support-consistent* ($\rightsquigarrow^*$), i.e., $x^1 \rightsquigarrow^* x^2 \dots \rightsquigarrow^* x^k$ in the scope of the BN $F_{[A,I]}$. The MSE of such a support-consistent Boolean state sequence X w.r.t. the time series T is noted $\widehat{\mathsf{mse}}(X, T)$; and the minimal distance among all support-consistent sequences in $F_{[A,I]}$ with T is referred to as $\widehat{\mathsf{mse}}(F_{[A,I]}, T)$. Because any reachable sequence is support-consistent (Theorem 1), we obtain that $\mathsf{mse}(F_{[A,I]}, T) \geq \widehat{\mathsf{mse}}(F_{[A,I]}, T) \geq \mathrm{MSE}_T$; and in particular $\mathsf{mse}(F_{[A,I]}, T) \neq \widehat{\mathsf{mse}}(F_{[A,I]}, T)$ only if none of the support-consistent sequences X with minimal $\widehat{\mathsf{mse}}(X, T)$ are actually sequences of reachable Boolean states. In such cases, F is a *false positive*. According to this definition, a BN is called *true positive* if one of the two following conditions hold. If $\widehat{\mathsf{mse}}(F_{[A,I]}, T) = \mathrm{MSE}_T$, then $\eta(T)$ is a valid sequence of reachable states in $F_{[A,I]}$. Otherwise, there exists at least one sequence X of reachable states such that $\mathsf{mse}(X, T) = \widehat{\mathsf{mse}}(F_{[A,I]}, T)$. Determining if F is a true positive can be done *a posteriori* with a model-checking approach.

Example Let us consider the Boolean network $F = (f_1, f_2, f_3)$ with $f_1(x) = x_1 \vee (\neg x_2 \wedge x_3)$, $f_2(x) = \neg x_1 \wedge \neg x_3$, and $f_3(x) = x_3 \vee (\neg x_1 \wedge x_2)$. Following the concrete semantics, from the state $(0,0,0)$, only the following transitions are possible:

$$(0,0,0) \to (0,1,0) \to (0,1,1) \to (0,0,1) \to (1,0,1)$$

With the meta-state semantics, the complete set of transitions from the meta-state corresponding to $(0,0,0)$ is the following:

$$(0,0,0) \rightrightarrows (0,\{0,1\},0) \rightrightarrows (0,\{0,1\},\{0,1\}) \rightrightarrows (\{0,1\},\{0,1\},\{0,1\})$$

Therefore, we obtain that $(0,0,0) \rightsquigarrow^* (1,1,1)$ (Definition 1) whereas $(0,0,0) \not\to^* (1,1,1)$: $(1,1,1)$ is not reachable from $(0,0,0)$ but it is support-consistent.

Now, let us consider the time series $T = \big(t^1 = (0.1, 0, 0.2), t^2 = (0.8, 0.9, 1)\big)$ and the support-consistent Boolean state sequence $X = \big(x^1 = (0,0,0), x^2 = (1,1,1)\big)$. We obtain $\widehat{\mathsf{mse}}(X,T) = \sqrt{0.1^2 + 0.2^2 + 0.2^2 + 0.1^2} = \mathrm{MSE}_T \approx 0.3$, which is also the minimal distance among all support-consistent sequences in F: $\widehat{\mathsf{mse}}(F,T) = \widehat{\mathsf{mse}}(X,T) \approx 0.3$. However, because $x^1 \not\to^* x^2$, the concrete sequence of reachable state the closest to T is $((0,0,0),(1,0,1))$. It results that $\mathsf{mse}(F,T) = \sqrt{0.1^2 + 0.2^2 + 0.2^2 + 0.9^2} \approx 0.9$. Therefore, in this example, $\mathsf{mse}(F,T) > \widehat{\mathsf{mse}}(F,T) = \mathrm{MSE}_T$ and F is a *false positive* Boolean network for T.

Optimization problem. We consider a PKN $\mathcal{G}$ and a set of r multiplex time series $D = (T^1, A^1, I^1), \ldots, (T^r, A^r, I^r)$. The distance between a BN F and the dataset is the sum of distances $\widehat{\mathsf{mse}}(F,D) \triangleq \sum_{l=1}^{r} \widehat{\mathsf{mse}}(F_{[A^l,I^l]}, T^l)$. The optimization procedure identifies the BNs compatible with the PKN $\mathcal{G}$ that have the minimal distance $\widehat{\mathsf{mse}}(F,D)$. In the scope of this paper, we enforce that each unobserved node whose value is not forced starts with the same initial value in all the trajectories: for all node $i \in \{m+1, \ldots, n\}$, there exists $x_i^0 \in \mathbb{B}$ such that for each $l \in \{1, \ldots, r\}$ with $i \notin A^l \cup I^l$, if X^l is a sequence of support-consistent Boolean states in $\mathbb{B}^n$ with $\widehat{\mathsf{mse}}(F_{[A^l,I^l]}, T^l) = \mathsf{mse}(X^l, T^l)$, then $X^l_{1,i} = x_i^0$. Whereas this constraint reduces the space of sequences to explore, it also ensures consistency between the different experimental settings.

Our method optimizes the MSE of the BNs F to the dataset D up to the reachability over-approximation criteria: if the BN is a true positive, the estimated MSE $\widehat{\mathsf{mse}}(F,D)$ is the exact MSE $\mathsf{mse}(F,D)$, otherwise the estimated MSE is an under-approximation - the exact MSE may be larger. As we return only the optimal solutions, all the BNs have the same estimated MSE.

MSE computation. Let us consider the distance between the squared MSE of a support-consistent sequence X with the minimum MSE_T:

$$\mathsf{mse}(X,T)^2 - \mathrm{MSE}_T^2 = \sum_{j=1}^{k} \sum_{i=1}^{m} ((x_i^j - t_i^j)^2 - (\eta(t_i^j) - t_i^j)^2)$$

Let us consider any i, j such that $x_i^j \neq \eta(t_i^j)$: remarking that $x_i^j = 1 - \eta(t_i^j)$, we obtain:

$$\begin{aligned}(x_i^j - t_i^j)^2 - (\eta(t_i^j) - t_i^j)^2 &= (1 - \eta(t_i^j) - t_i^j)^2 - (\eta(t_i^j) - t_i^j)^2 \\ &= (1 - 2\eta(t_i^j))(1 - 2t_i^j) = |1 - 2t_i^j| = 2|t_i^j - 0.5|\end{aligned}$$

Therefore, the minimization of $\mathsf{mse}(X,T)$ can be achieved by minimizing the number of data points i, j such that $x_i^j \neq \eta(t_i^j)$, weighted by the distance of the data point to the threshold ($|t_i^j - 0.5|$). Furthermore, the optimization can be done with respect to a relative MSE, without necessarily computing the absolute value of the MSE. In such a case, one can still compute the absolute value of the estimated MSE *a posteriori* using the above equation on a single BN sampled from the result set with one Boolean trace X for each time series T of dataset D such that $\widehat{\mathsf{mse}}(X,T)$ is minimal.

Minimal solutions. Depending on the number of nodes in the PKN, and on the discriminative power of the time series dataset, a rather large number of BNs may be expected to be inferred. As an alternative, we can output only the BNs having the smallest Disjunctive Normal Form (DNF) representation with respect to clause inclusion, i.e., no literal nor clause can be removed. This means that no unnecessary edges occur in the BNs, thus providing only the simplest BNs. In the following, we refer to such a set of solutions as *subset-minimal*. Similarly, only the BNs having the smallest number of literals in their DNF representation can be considered: those solutions, which are also subset-minimal, are referred to as *cardinal-minimal*.

2.4. Implementation

Answer Set Programming (ASP; [15, 16]) is a declarative approach to solving knowledge-intense combinatorial (optimization) problems comprising up to tens of millions of variables. ASP's distinguishing combination of a high-level modeling language with high-performant solving tools allows for concentrating on an actual problem, rather than a smart way of implementing it. The basic idea of ASP is to express a problem in a logical format so that the (logic) models of its representation provide the solutions to the original problem. Problems are expressed as logic programs and the resulting models are referred to as answer sets. Although determining whether a program has an answer set is the fundamental decision problem in ASP, modern ASP solvers like *clasp* [17] support various combinations of reasoning modes, among them, regular and projective enumeration, intersection and union, multi-criteria optimization and subset [18] and/or sum-based minimal (maximal, resp.) model enumeration. We elaborate on the representation of our problem in ASP in Appendix A.

The output of the ASP encoding is the set of optimal BNs with, for each BN, a minimal set of time-points of the time series data which do not verify the support-consistency in the associated BN (errors). When no errors are returned, $\widehat{\mathsf{mse}}(F, D) = \mathrm{MSE}_D$, otherwise $\widehat{\mathsf{mse}}(F, D) > \mathrm{MSE}_D$, where $\mathrm{MSE}_D \triangleq \sum_{l=1}^{r} \mathrm{MSE}_{T_l}$.

2.5. A posteriori *filtering of true positives with model-checking*

As explained in the previous sections, the set of Boolean networks returned by our method is an over-approximation: there may be false positive answers, whereas it is guaranteed that no false negative exists. Recall that a BN is *true positive* if, for each time series T with forced activations A and forced inhibition I, either $\widehat{\mathsf{mse}}(F_{[A,I]}, T) = \mathrm{MSE}_T$ and $\eta(T)$ is a valid sequence of reachable (sub)states in $F_{[A,I]}$; or $\widehat{\mathsf{mse}}(F_{[A,I]}, T) > \mathrm{MSE}_T$ and there exists at least one sequence X of reachable states in $F_{[A,I]}$ with minimal $\widehat{\mathsf{mse}}(X, T)$. In that latter case, the sequence X is constructed from the original sequence $\eta(T)$ corrected following the errors identified by our encoding, i.e., the time-points which do not verify the support-consistency.

Therefore, one can verify if a returned BN is actually a true positive by verifying if it satisfies adequate reachability properties. Because of the non-deterministic dynamics of BNs, we use the *Computation Tree Logic* [19] to express the reachability properties. Basically, a sequence $X = (x^1, x^2, \ldots, x^k)$ with $\forall j \in \{1, \ldots, k\}, x^j \in \mathbb{B}^n$ is interpreted as the property: from state x^1, there exists a sequence of transitions leading to x^2; followed by a sequence of transitions leading to x^3, etc. This is expressed in CTL by nested $\mathtt{EF}$, where $\mathtt{E}$ stands for the existence of an execution, and $\mathtt{F}$ for the eventually operator: $x^1 \Rightarrow \mathtt{EF}(x^2 \wedge \mathtt{EF}(x^3 \wedge \ldots))$.

2.6. Application to the toy example

Let us illustrate now the application of the method to our toy example in different configurations. We assume that we are given the PKN shown in Fig. 1 and several time series, and apply our Boolean network identification method.

Let us first consider the two time series corresponding to the two left-most traces in Fig. 1c. The time series are composed of only two states, but note that the fact that the last state is stable is forgotten here: our method makes no hypothesis on the stability of the last observed time point. Therefore, we will consider all the networks that can reach the second state, but do not enforce that this latest state is a fixed point. All the nodes are observed, and the time series have different experimental conditions: in the first trace, the node nI is forced to be constantly inactive, whereas nJ is forced to be constantly active ($A = \{nJ\}, I = \{nI\}$); in the second trace, the node nI is forced to be constantly inactive, whereas nJ is free ($A = \emptyset, I = \{nI\}$).

In such a setting we can identify 12 Boolean networks (Fig. 3(a)) that are compatible with the given PKN and that can reproduce the two traces, i.e. the two reachability constraints w.r.t. the perturbation settings. Among the identified networks (all true positives), the network in Fig. 1a is obviously present. In addition, networks without any circuit in their interaction graph are identified. Therefore, one can conclude that a feedback circuit between nodes nA and nB is not necessary to reproduce those two time series.

Let us now consider the four time series of Fig. 1c. Only 2 Boolean networks (Fig. 3(b)) are identified from the given PKN and those time series with perturbation. Both networks contain a (negative) circuit between nA and nB, hence stressing the necessity of this circuit to reproduce the time series.

In the above settings, the value of nodes in the considered time series are either 0 or 1. Therefore, the MSE of the identified Boolean network is 0. Let us now consider the same dataset of four time series, where an additional observation is appended to the first time series:

$$\begin{array}{l} nI : \text{not act.} \\ nJ : \text{act.} \\ nA : \text{init} \\ nB : \text{init} \end{array} \quad \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0.4 \end{pmatrix}$$

For such a time series T, $\text{MSE}_T = 0.4$. It appears that none of the Boolean networks compatible with the PKN can reproduce this new set of time series. Indeed, in the two networks identified in the previous settings, the state $(0, 1, 0, 1)$ is a fixed point, therefore, it is impossible to reach the binarized state $(0, 1, 0, 0)$ from it. Nevertheless, the two same networks are the ones showing the minimal MSE from the new data: 0.6, which corresponds to MSE_T increased by twice the distance of the observation 0.4 to the threshold 0.5 (see the last paragraph of Section 2.4)

3. Effectivity of the learning procedure

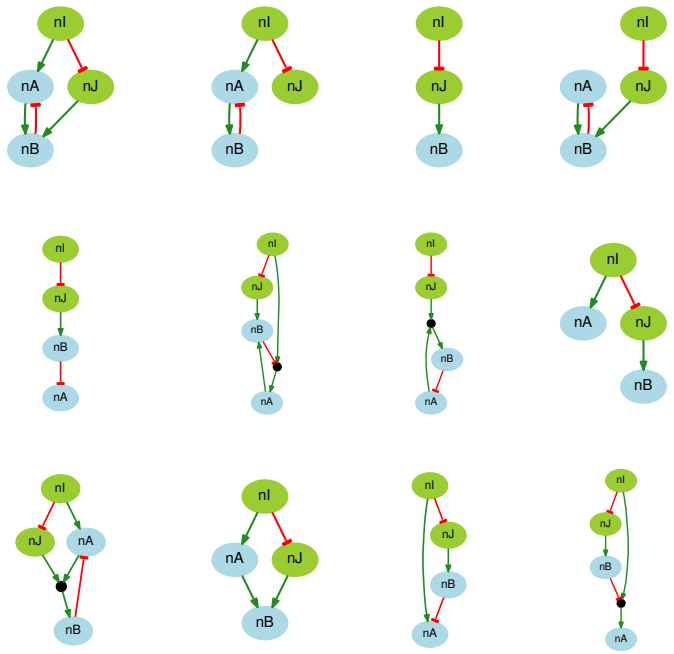
3.1. Testing models

In order to assess the applicability of our method to networks with different topologies, we applied our approach to identify the BNs for the PKN of 3 different litterature-based signaling models with different characteristics in terms of size and number of logical gates they may lead to. In cases where the signaling model was fully determined, the PKN we used was its compatible IG which corresponded to its network structure.

For each PKN, synthetic 10 time-point time series data were generated by simulating logic-based ODEs associated with three randomly selected golden-standard BNs whose IG was compatible with the considered PKN. The logic-based ODEs compuation were done using the CNORode R package, which implemented the method described in [20]. The randomly selected BNs had equal probability given the full pool of comaptible BNs.

- (Case-Study A) The first one is the EGF-TNFα signaling pathway published in [5] and illustrated in Fig. 6A. We generated 10 perturbation conditions containing synthetic time-series data for the BNs associated to this PKN.
- (Case-Study B) The second is the PKN of the TCR signaling model [21]. From this PKN several versions were derived by choosing different sizes of graphs and choices for the experimental-design (sets of fixed stimuli, inhibitors and species with time-series measures associated). With a first experimental design, a subgraph of 14 nodes and 22 edges was generated (Case-Study B.1), see Fig. 4. With a distinct experimental design, a compressed subgraph (Case-Study B.2), composed of 16 nodes and 25 edges, and the complete PKN (Case-Study B.3), composed of 40 nodes and 58 edges, were generated. 10 perturbation conditions of synthetic time-series data were simulated for the BNs associated to these PKNs.
- (Case-Study C) The last one was the PKN of the ERBB receptor-regulated G1/S transition model [22]; we used the compressed version of this PKN. 10 perturbation conditions of synthetic time-series data were simulated for the BNs associated to this PKN.

(a)



(b)

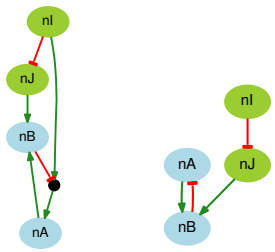


Figure 3: (a) Boolean networks identified from the two left-most time series of Fig. 1c; (b) Boolean networks identified from the four time series of Fig. 1c.

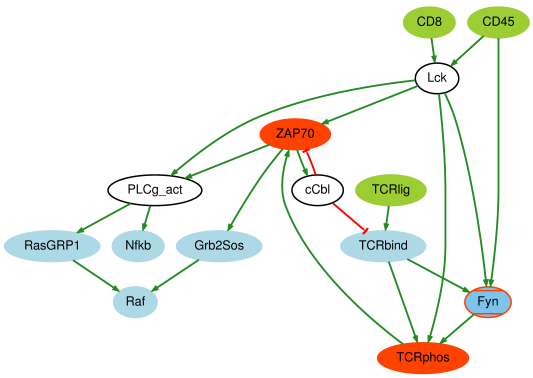


Figure 4: PKN representing TCR activation extracted from [21]. Green and red edges indicate activations and inhibitions respectively. Colors of the nodes represent the chosen experimental design: green refers to inputs/stimuli, red, to inhibited nodes, and blue, to measured species. In total 10 perturbations were simulated from 3 BNs extracted from this PKN.

In order to evaluate the impact of our method, they were ordered according to the size of the search space of boolean models they generate. Notice that the search space increases exponentially with the number of edges, specifically with the number of predecessors of a node. As shown in Table 1, according to this criteria, the Case-Study C is much more complex that Case-Study B.3 although it has less nodes and edges. This is due to the fact that Case-Study B.3 is an uncompressed network with less candidates for boolean gates than Case-Study C.

The data files, implementation, and scripts for reproducing the results of this section can be downloaded from https://github.com/pauleve/caspots.

3.2. Computation time

For each dataset, the model identification has been performed with respect to the PKNs from which the BNs used for synthetic data generation have been extracted. We distinguished two optimization modes: (i) the identification of BN with minimal MSE and minimal cardinality and (ii), the identification of BN with minimal MSE and such that any sub-model has no more a minimal MSE (called subset-minimal).

In Table 1, we can see that obtaining one optimal solution takes few seconds in both optimization modes, while enumerating the full list of optimal solutions takes longer and also scales with the number of solutions. This is illustrated, for instance, in the second experiment of Case-Study C: the two cardinal-minimal solutions were identified in 3s (the second solution is outputted immediately after the first), whereas it took almost 2 additional minutes to prove that the enumeration is exhaustive. In all the case-studies it was possible to retrieve the exhaustive set of BNs with minimal MSE and minimal cardinality (which are also subset-minimal). This shows that our method can detect optimal BNs for large networks of over 50 edges with a search space of 2^{174} compatible BNs with the PKN.

In the larger cases, TCR signaling (Case-study B.3) and ERBB (Case-study C), the number of subset minimal solutions goes beyond 100,000 and that is the reason why we stopped the computation (after 3h).

4. Soundness of the learning procedure

4.1. Soundness of the case-study BNs

Following Section 2.5, we perform an *a posteriori* analysis of the solutions returned by our method to compute the proportion of the BNs that are true positives (TP). We performed the model-checking when the set of BNs was below 100,000 networks using the symbolic model-checker `NuSMV` [23]. The results are displayed in Table 1. The model-checking of one BN took between 1s and 10min. Due to the number of nodes in the uncompressed version of the TCR signaling model (Case-Study B.3), the exact model-checking of identified BNs was not tractable using `NuSMV`. In such cases, one could consider using approximate model-checking methods (e.g., bounded model-checking [24]) and/or model-reduction [25].

		cardinal-minimal			subset-minimal		
Model	Space	First	Total	TP	First	Total	TP
Case-Study A		<1s	8 (1s)	100%	< 1s	54 (2s)	100%
TNFα-EGF [5]	2^{21}	1s	12 (5s)	100%	1s	64 (3s)	100%
13 nodes, 16 edges		<1s	4 (1s)	100%	<1s	36 (3s)	100%
Case-Study B.1		1s	18 (5s)	100%	1s	5,544 (3min)	100%
TCR signaling [21]	2^{37}	1s	2 (5s)	100%	1s	2,901 (90s)	100%
14 nodes, 22 edges		1s	8 (5s)	100%	1s	6,510 (4min)	100%
Case-Study B.2		2s	4 (12s)	100%	1s	73,962 (1h40)	100%
TCR signaling [21]	2^{49}	3s	4 (25s)	0%	1s	68,338 (1h30)	78%
16 nodes, 25 edges		3s	20 (23s)	90%	1s	74,757 (1h40)	96%
Case-Study B.3		4s	8 (90s)	-	5s	>100,000	-
TCR signaling*[21]	2^{106}	6s	8 (90s)	-	5s	>100,000	-
40 nodes, 58 edges		4s	8 (60s)	-	5s	>100,000	-
Case-Study C		7s	19 (7min)	42%	6s	>100,000	-
ERBB [22]	2^{174}	3s	2 (2min)	100%	5s	>100,000	-
19 nodes, 50 edges		5s	69 (6min)	19%	5s	>100,000	-

Table 1: Learning method applied to 5 PKNs. For each PKN 3 compatible Boolean Networks were used to generate 3 sets of synthetic time series data. The search space column corresponds to the number of possible Boolean functions which can be selected and that are compatible with the PKN. The min-cardinal and min-subset column show the results obtained when the optimization criteria was set to find the minimal cardinality and minimal subset solutions respectively. The First column corresponds to the computation time of the first optimal solution. The Total column corresponds to the number of all optimal solutions and its computation time (in parenthesis). The TP column corresponds to the rate of true positives, when applicable.

For Case-Studies A and B.1, it turns out that no false positive where returned by our method. For Case-Studies B.2, the subset-minimal solutions also show a very high rate of true positive. This supports that our over-approximation criteria, i.e., the support consistency in the meta-state semantics (Definition 1), can discriminate efficiently among the BNs in the search space.

One can remark that for Case-Study B.2 and C the true positive rate of cardinal-minimal solutions is low for some datasets, especially compared to the true positive rate of subset-minimal solutions for the second dataset of Case-Study B.2. This can be explained by the fact that cardinal-minimal solutions are somehow the smallest solutions that satisfy the necessary (over-approximation) criteria, which can turn out to be not enough to satisfy the concrete reachability properties.

4.2. Impact of the over-approximation criteria for reachability

In order to have a better estimation of the impact of the over-approximation depicted in Section 2.2, a benchmark of 10 additional PKN's were derived by randomly removing or adding edges from the TNFα-EGF model (Case-Study A). These PKN's are named Case-Study A.1, ... A.10.

For each PNK A.i, 3 random consistent BN were generated, each allowing the generation on a synthetic dataset (dataset A.i.1, A.i.2, A.i.3). The learning algorithm was applied to the PKN associated to the 3 synthetic datasets and enabled the identification of subset-minimal BN with minimal MSE. As depicted in Table 2, the estimated minimal MSE was equal to the minimum MSE_T for all 30 synthetic datasets. In addition, in 28 cases (over 30), 100% of the BNs reported by the identification algorithm reproduced the synthetic boolean traces and therefore were true-positive BNs. In only two cases of the 30 experimentations (datasets A.2.3 and A.5.1), respectively 9% and 2% of the identified BNs were not capable of reproducing the boolean traces.

Additional experiments were performed to investigate the impact of incomplete knowledge over the identification of BN's. For each PKN A.i, a degraded PKN with 1 deletion was produced, as well as a degraded PKN with 2 deletions. Then the identification procedure was applied to the (incomplete) PKNs with 1 and 2 deletions with respect to the initial datasets associated to A.i. Contrary to the previous experiments, in these cases, there was no insurance that there even exists a BN reporting the experimental datasets. With the PKNs with deleted edges, most of the cases show a very high true positive rate (often 100%) and an estimated minimal MSE close to (often equal to) MSE_T. Note that for some datasets, no true positive has been found. For the cases when the estimated minimal MSE is different from MSE_T, the true positive rate can only be evaluated by sampling Boolean traces close to the time series data. Because of the very high combinatorics of such sampling space, the computation has been aborted after one hour,

Dataset	Exact PKN			PKN w/ 1 deletion			PKN w/ 2 deletions		
	TP	#	ΔMSE	TP	#	ΔMSE	TP	#	ΔMSE
A.1.1	100%	54	0	100%	18	0	100%	6	0
A.1.2	100%	12	0	100%	9	0.06	100%	3	0.06
A.1.3	100%	24	0	100%	18	0.07	100%	6	0.07
A.2.1	100%	64	0	100%	16	0	100%	16	0
A.2.2	100%	264	0	100%	48	0	100%	12	0
A.2.3	98%	258	0	96%	54	0	100%	18	0
A.3.1	100%	36	0	100%	18	0	100%	6	0.07
A.3.2	100%	72	0	100%	36	0	100%	27	0.04
A.3.3	100%	72	0	100%	36	0	100%	27	0.04
A.4.1	100%	216	0	100%	72	0	100%	36	0
A.4.2	100%	504	0	100%	168	0	100%	48	0
A.4.3	100%	156	0	100%	60	0	100%	6	0
A.5.1	91%	675	0	≥0 %	135	≥0.04	≥0 %	54	≥0.03
A.5.2	100%	2640	0	100%	1188	0	100%	486	0
A.5.3	100%	780	0	≥0 %	156	≥0.01	≥0 %	36	≥0.01
A.6.1	100%	890	0	100%	285	0	100%	120	0
A.6.2	100%	2304	0	100%	720	0	100%	288	0
A.6.3	100%	426	0	100%	126	0	100%	102	0
A.7.1	100%	960	0	100%	456	0	100%	140	0
A.7.2	100%	820	0	100%	364	0	100%	144	0
A.7.3	100%	1536	0	100%	480	0	100%	144	0
A.8.1	100%	108	0	100%	54	0	100%	54	0
A.8.2	100%	54	0	≥0 %	54	≥0.23	≥0 %	54	≥0.23
A.8.3	100%	96	0	≥0 %	>2	≥0.19	≥0 %	>3	≥0.19
A.9.1	100%	21	0	100%	3	0	100%	3	0
A.9.2	100%	156	0	100%	54	0	100%	18	0
A.9.3	100%	36	0	100%	36	0	100%	12	0
A.10.1	100%	336	0	100%	84	0	100%	36	0
A.10.2	100%	864	0	100%	216	0	100%	72	0
A.10.3	100%	864	0	100%	216	0	100%	72	0

Table 2: Summary of true positive rate (TP), number of subset-minimal solutions (#), and distance to minimum MSE MSE_T, noted $\Delta\mathrm{MSE} \triangleq \mathsf{mse}(F, D) - \mathrm{MSE}_T$, for the EGF-TNF$\alpha$ dataset with the exact and incomplete PKNs. True positive rates pre-pended with "≥" indicates that the computations have been aborted due to time limit constraints, therefore the rate may be under-estimated. When no true positive have been found, the displayed ΔMSE may be under-estimated because $\mathrm{MSE}_T \leq \widehat{\mathsf{mse}}(F, D) \leq \mathsf{mse}(F, D) = \mathrm{MSE}_T + \Delta\mathrm{MSE}$; this is indicated by pre-pending ≥ to ΔMSE.

hence we cannot guarantee that no true positive exists. When no true positive was identified, the minimal MSE may be under-estimated.

The inference of the subset-minimal solutions for the 30 benchmarks with exact EGF-TNFα PKNs took less than 2 seconds on average. The performance is similar for the benchmarks with incomplete PKNs that contained a true positive BN in the result. The number of learned BNs varies between 12 and 2640 with the exact PKN and from 2 to 1188 with modified PKNs. Depending on the size and the complexity of its dynamics, the model-checking of one BN took between 1s and 5 minutes. The full set of solutions (not only the subset-minimal BNs) has also been performed with the exact PKN, showing very similar results and running time, with subsequently more results (up to 54,000 BNs, data not shown). Same experiments have been conducted on the time series generated with noise but show no difference in the results (data not shown). This may indicate that the noise influence may be tempered by the binarization.

5. Added-value of the identification method over the quality of learned BNs

5.1. Gain in terms of fitness with respect to pseudo steady-states methods

In this section we compare our results with the previously developed approach *Caspo* [26]. Caspo, as well as other state-of-the-art approaches such as CellNopt [27], considers two time-points (an initial point and a pseudo-steady state) and a PKN. It computes a set of BNs with minimal size that can explain the best the transition between the two time-points. Due to its static nature and the minimal size condition, it is not possible to infer feedback loops or dynamic behaviour, because models with loops would not improve the fitting with the data assuming a steady state and contain more gates or edges which render them not minimal in terms of size or subset minimality criteria. With this comparison, we aim at emphasizing the importance of taking into account model dynamics to obtain accurate model predictions.

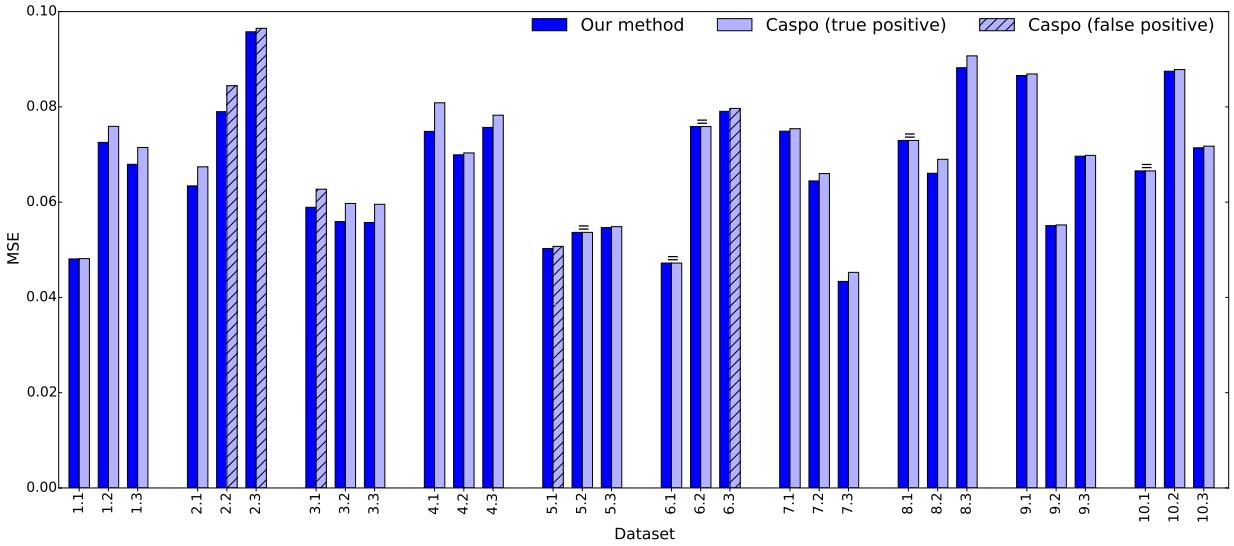


Figure 5: Comparing minimal MSE with *Caspo* for 10 different PKNs with 3 datasets each. "=" indicates equal minimal MSE.

We applied *Caspo* and our method on the 10 PKNs A.1, ... A.10 derived from the EGF-TNFα model, each being associated to its three synthetic datasets introduced in section 4.2. We compared the minimal MSE obtained applying our optimization procedure on the BNs returned by *Caspo* on one time point (assumed steady by *Caspo*) and on all BNs delimited by the PKNs. Therefore, we compare the estimated minimal MSE with respect to the multiplex time series for both the *Caspo* approach and the method introduced by this paper. As explained in Section 2.3 the computed MSE may be under-estimated so we used model-checking *a posteriori* to verify the presence of true positive BNs.

Fig. 5 plots the estimated minimal MSEs, where the 6th time point of the time series has been selected for the learning with *Caspo*; other time points give very similar results (data not shown). On the contrary to our approach where it was always possible to find a BN which was fully consistent with the data, having the minimum $MSE = \text{MSE}_T$, *Caspo* failed to identify a consistent model with the data in 25 over the 30 experiments. Among those 25 experiments, the estimated MSE on *Caspo* results may be under-estimated in 5 experiments where the returned BNs are actually false positives (streaked bars). This evidences the role of feedback loops which cannot be captured with a two-timepoints learning procedure and the information gain brought by time series data.

5.2. Application to error detection

In order to have a last and further insight of the biological added-value of our method, we applied the method to a final dataset of perturbation data. To that goal, we studied an example introduced in [5] to model a perturbated EGF-TNFα signaling network (Case-Study A). To that goal, the synthetic data depicted in Fig. 6C were produced from an admissible golden-standard BN for this PKN. Then, a degraded PKN was obtained by removing the link from tnfa to ap1 from the PKN to represent incomplete regulatory knowledge.

After confronting the incomplete PKN with the time series data, our method learned a family of BNs consisting of 3 subset-minimal BNs. All BNs were checked to be true positives, therefore they have an optimal ΔMSE ($\Delta\text{MSE} \triangleq \mathsf{mse}(F, D) - \text{MSE}_T$) score of 0.07 with respect to the data.

The family of optimal BNs was learned after 0.04 seconds of computation on a standard desktop computer. This corresponds to a 2 to 4 orders of magnitude improvement of the computation time.

In Fig. 6B we plotted the subset-minimal family of BNs learned for this case study. It recovers the complete logical behaviors of the golden-standard, except for the one regulation from tnfa to ap1 which was

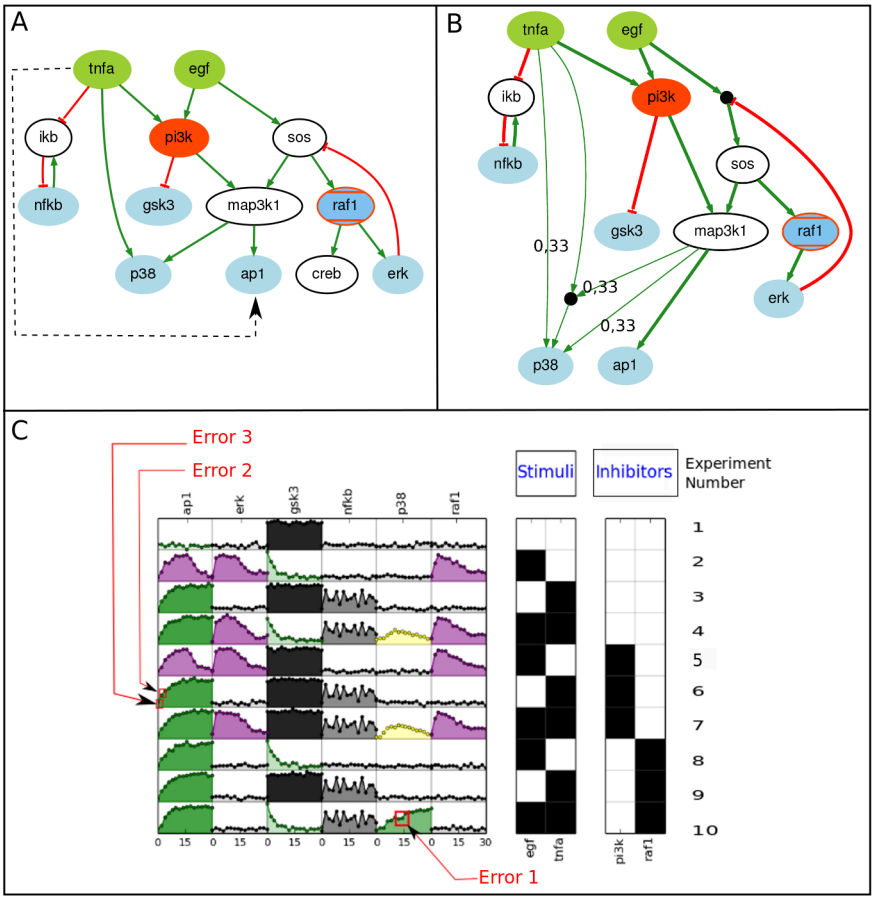


Figure 6: **(A)** Compressed PKN from [5]. Green and red edges indicate activations and inhibitions respectively. The dashed black edge was removed from the PKN prior learning to simulate learning BNs with uncomplete prior knowledge. Colors of the nodes represent the chosen experimental design: green refers to inputs/stimuli, red, to inhibited nodes, blue, to measured species, and blue nodes lined in red mean both measured and inhibited species. **(B)** Boolean networks (BNs) learned from time series data which are subset-minimal. All BNs predictions have minimal ΔMSE with respect to the synthetic time series data. A black circle represents a logical AND gate. A number written over an edge represents the frequency of this logical gate or edge with respect to the family of BNs when the edge is not shared by all BNs. **(C)** Synthetic time series data used in [5] simulated using a BN admissible for the PKN in A. In total 10 experimental conditions were simulated. Red boxes indicate the minimal set of 3 specific time-points where the traces could not be reproduced according to the BN dynamics using the over-approximated reachability.

removed from the PKN. Only the logical function of the regulation over p38 is not consensually learned across all BNs in the family; the rest of logical functions learned are shared by all models.

The quality of our results concerning the learned BNs is comparable to the one obtained in [5] for the same case study. Moreover, our method is exhaustive: all logical networks are learned. The full set of solutions (not only the subset-minimal BNs) was also computed showing one more BN with an OR gate above p38 from tnfa and map3k1.

The method also automatically identified the list of minimal errors in the time series data, selecting time-points that cannot be explained by the learned BNs. For the case of all optimal BNs, we found the following 3 errors (see Fig. 6C) in all of them. For *experiment 10, time-point 10, species p38,* the error can be explained by the noise artificially introduced in the dataset. The predecessors of p38 are tnfα and egfr, both active in experimental condition 10 (see Fig.6C). The signal of p38 can therefore only increase (or stay the same). However, the measure of p38 slightly decreases (due to noise) at time-point 10; this generates an error since the BNs cannot satisfy the data at this particular time-point. For *experiment 6, time-point 2 and 4, species ap1,* the errors can be explained by the fact that one edge (the link from tnfα to ap1)

was deleted from the PKN, but was kept to generate the synthetic time series data. All BNs agree on a regulation of p38 and ap1 from map3k1. In experiment 6 tnfα is stimulated and pi3k is inhibited (see Fig. 6C). At time-point 2 the value of map3k1 has to be activated (transition $0 \rightarrow 1$) to justify the activation of ap1. However, since map3k1 is the only regulator of p38, which is all the experiment at value 0, this cannot be explained by the BN and generates an error.

6. Conclusion

We have introduced a procedure based on combinatorial optimization with declarative programming approaches and model checking to identify BNs from multiplex time series data given a prior network structure. To cope with the complexity of an exhaustive analysis of BNs dynamics, we defined an abstract semantics of BNs from which we derived a necessary condition for the satisfaction of successive reachability properties, induced by the time series data. Our procedure identifies all the BNs that satisfy this necessary condition with the shortest distance (in terms of MSE) to the observed experimental data. Because the satisfaction criteria for the dynamics is over-approximated, our method may lead to BNs that are false positive, and have an under-estimated MSE. Applied to synthetic multiplex time series datasets on networks composed of 13-40 nodes and 16-50 edges, the identification of one optimal BN takes only a few seconds, showing a remarkable efficiency. In addition, the rate of true positives is over 78% for networks having less than 25 edges.

In the present form, we assume that the experimental data is normalized between 0 and 1 and use a discretization threshold at 0.5. Whereas such a setting is relevant for phosphoproteomics data, future work may generalize our optimization framework to account for adaptive and multiple discretization levels. Moreover, application to larger networks should be considered, although few of such data are currently available, and generating synthetic data with sufficient discriminant power may be challenging. Indeed, the presence and nature of edges is deduced from time series data, and the larger the network, the harder to have enough data to identify a large proportion of interactions.

Because our identification method can be exhaustive, the framework we propose is suited for the complete *Thomas parameters identification* for BNs from incomplete time series data [28, 29].Thanks to our abstract semantics, our method is able to filter out very efficiently a large number of candidate BNs without a costly exact model-checking, which is postponed to the validation of the results. In that way, future work may further explore the combination of dynamics over-approximations with model-checking approaches to provide scalable and exact inference of BNs from time series data.

Appendix A. ASP Encoding

At first, we will describe the input format of the PKN and the time series data that is required by our encoding. A PKN is described by the functions of its admissible BNs. The predicate `node/1` describes a set of nodes. For each node N, we have a set of functions f_N, described by the predicate `fun(F,N,I)` where `F` is the number of inputs of the function, `N` is the node id and each `I` is an identifier for an input of the function. An input is described by `in(I,N,S)` where `I` is the identifier of the input, `N` is the input node and `S` is either 1 or -1, standing for a positive or negative input. If we have several inputs with the same id for one function (like `fun(2,nB,4)` in Listing 1) this is meant to be a conjunction of inputs. With the set of facts in Listing 1, we can describe our example PKN in Fig. 1a.

```
node(nI). node(nJ). node(nB). node(nA).
fun(1,nJ,1). in(1,nI,-1).
fun(1,nB,2). fun(1,nB,3). in(2,nJ,1).  in(3,nA,1).
fun(2,nB,4). in(4,nJ,1).  in(4,nA,1).
fun(1,nA,5). fun(1,nA,6). in(5,nI,1).  in(6,nB,-1).
fun(2,nA,7). in(7,nI,1).  in(7,nB,-1).
```

Listing 1: Toy Example PKN

We describe a set of experiments with a set of facts (see Listing 2). Predicate `obs(E,T,N,V)` represents the value `V` of the node `N` in experiment `E` and timestep `T`. The value is expressed in hundredth and is rounded (0.346 will be encoded as 35). To describe a clamping of a node we use `clamped(E,N,V)` to say that node `N` was clamped to `V`. A sample measurement is given in Listing 2.

```
obs(ex1,0,nI,12). obs(ex1,0,nB,07). obs(ex1,0,nA,14). clamped(ex1,nJ,1).
obs(ex1,1,nI,27). obs(ex1,1,nB,58). obs(ex1,1,nA,13).
obs(ex1,2,nI,15). obs(ex1,2,nB,87). obs(ex1,2,nA,06).
```

Listing 2: Toy Example Data

```
1  % enumerate all formulas compatible with the pkn for each node
2  {dnf(N,I) : fun(F,N,I)} :- node(N); fun(_,N,_).
3  clause(I,N,S)           :- in(I,N,S); dnf(_,I).

5  % contradictions
6  :- dnf(N,I); in(I,U,S); in(I,U,-S).

8  % J2 subsumed by J1
9  :- fun(F1,N,I1); fun(F2,N,I2); F1 < F2;
10    dnf(N,I1);  dnf(N,I2); clause(I2,U,S) : clause(I1,U,S).
```

Listing 3: Admissible Boolean Networks

For the encoding we follow the general design approach in ASP in a way that we first describe the search space for all admissible BNs given a PKN. In the context of ASP this does not mean choosing a BN by some heuristic, but exhaustively trying all possible combinations of edges and logical connectives, and only allowing these combinations that are consistent with the given encoding. Afterwards, we specify restrictions (integrity constraints) on this set of admissible BNs (namely that they have to be compatible with the data).

An encoding consists of a set of rules. Informally, a rule means that if all atoms in the body of a rule are true (everything right of "`:-`") the head (the atom left of "`:-`") is inferred to be true. If there is no head given, this is called an integrity constraint. It is not allowed that all atoms in its body are true. Furthermore, we will use choice rules in our encoding, depicted by the curly braces in the head. These allow us to exhaustively test all combinations of atoms in the head to be true. We use this to describe the search space, e.g. all possible admissible BNs. We refer the interested reader to [30] for a detailed description of the syntax of ASP. In Listing 3, for each node we chose a set of functions to be in a disjunction in Line 2 and 3. In Line 6, we exclude all BNs that have contradictory inputs. Now only BNs admissible with the PKN will be generated. We furthermore avoid producing irrelevant BNs by not allowing functions that are subsumed by others in Lines 9-10.

```
1  timepoint(E,T) :- obs(E,T,_,_).
2  obs(E,T,N) :- obs(E,T,N,_).
3  obs(E,T,N) :- clamped(E,N,_); timepoint(E,T).
4  unobs(E,T,N) :- timepoint(E,T); node(N); not obs(E,T,N).

6  measured(E,T,N,1) :- obs(E,T,N,X); X >= 50.
7  measured(E,T,N,0) :- obs(E,T,N,X); X < 50.
8  1{measured(E,T,N,(1;0))}1 :- unobs(E,T,N).
```

Listing 4: Discretizing

After creating an encoding to produce admissible BNs, we now describe how we normalize the measured data (Listing 4). The first four lines create sets of timepoints, observed and unobserved nodes. Afterwards, we discretize the observed data values to Boolean values in Lines 6 and 7. As some nodes may be unobserved, we use a choice rule to exhaustively test all possible values for them.

```
1  1{guessed(E,T,N,(1;0))}1  :- obs(E,T,N,_).
2  1{guessed(E,T,N,(1;0))}1  :- unobs(E,T,N).

4  :- unobs(E,T,N); measured(E,T,N,V); not guessed(E,T,N,V).
5  :- unobs(E1,0,N); unobs(E2,0,N); E1 < E2;
6     measured(E1,0,N,V); not measured(E2,0,N,V).

8  error(E,T,N) :- measured(E,T,N); not guessed(E,T,N).
```

Listing 5: Guessing Data

As there might not exist a BN that is consistent with the discretized, measured data (`measured/4`), we simply create a new set of data by exhaustively enumerating all possible measurements. Therefore, we again use a choice rule to enumerate a Boolean value for every node (Line 1 and 2 in Listing 5). We call this data "guessed" data. The BNs we compute will be support consistent with respect to this guessed data. If the guessed data is the same as the discretized measured data, then the MSE of the BN is the minimum MSE_T. To relate the guessed data (which can be any combination of values) to the actual discretized measured data, we will minimize the difference between the guessed and the measured data, later on. As it does not make sense to "guess" the value of an unobserved node, Line 4 ensures that the fake measurement of it coincides with the guessed value. Furthermore, we have the precondition that the value of the unobserved nodes is the same among all experiments in the first timestep of the experiments (Line 5 and 6). The difference between the guessed and the measured data, can directly be read of the error predicate in Line 8.

We now move on to describe the support consistency condition (Listing 6), which will remove all admissible BNs that are not support consistent with the guessed data.

```
1  metaU(E,T,N,V) :- guessed(E,T,N,V).
2  metaU(E,T,N,V) :- clamped(E,N,V); timepoint(E,T).

4  convert(-1,0).
5  convert(1,1).
6  ptrans(E,T,clause,X,1) :- metaU(E,T,N,C) : clause(X,N,V), convert(V,C);
7                            timepoint(E,T); clause(X,_,_).
8  ptrans(E,T,clause,X,0) :- metaU(E,T,N,0); clause(X,N,1).
9  ptrans(E,T,clause,X,0) :- metaU(E,T,N,1); clause(X,N,-1).

11 ptrans(E,T,dnf,X,1) :- trans(E,T,clause,C,1); dnf(X,C).
12 ptrans(E,T,dnf,X,0) :- trans(E,T,clause,C,0) : dnf(X,C); dnf(X,_).

14 clamped(E,N) :- clamped(E,N,_).
15 {trans(E,T,N,X)} :- ptrans(E,T,dnf,N,X); not clamped(E,N).

17 metaU(E,T,N,X) :- trans(E,T,N,X).

19 opposite(1,0).
20 opposite(0,1).
21 :- guessed(E,T+1,S,V); not metaU(E,T,S,V).
22 :- guessed(E,T+1,N,Y); opposite(X,Y); trans(E,T,N,X); not trans(E,T,N,Y).
```

Listing 6: Support Consistency Check

The first thing we want is to exhaustively enumerate all possible meta-states $u \in \mathbb{M}^n$ with $x \rightrightarrows^* u$ as in definition Lemma 1. Therefore, we include the current state x in u in the first two lines. In Lines 4-12, we derive all values that we can potentially reach by a transition from the state u. In Line 15, we now enumerate to make a transition or not, and if yes, add it to our meta state in Line 17. Given this meta state, we can now easily check the support consistency condition. Line 21 ensures that the next state y belongs to the meta-state u ($y \in u$). Line 22 checks the second condition, that if for some $i, x_i = y_i$ and $\{0, 1\} = u_i$ then there exists a transition $f_i(z) = y_i$.

Recall that in case where the guessed data coincides with the discretized measured data, the BN reproduces the time series data, and the MSE is minimal. As there exist PKNs where it is not possible to find such a BN, we want to find the BNs with the smallest MSE possible. Therefore we minimize the difference

```
1 #minimize{Erg@2,E,T,S : measured(E,T,S,V), not guessed(E,T,S,V),
2                       obs(E,T,S,M),Erg=50-M, M < 50}.
3 #minimize{Erg@2,E,T,S : measured(E,T,S,V), not guessed(E,T,S,V),
4                       obs(E,T,S,M),Erg=M-49, M >= 50}.
5 #minimize{F@1,N,I : dnf(N,I), fun(F,N,I) }.
```

Listing 7: Minimizing MSE and Size

between the guessed and the measured data in Lines 1-4 in Listing 7. The last line gives the secondary optimization goal, and is used to either compute all cardinality or subset smallest solutions[2]. The minimization with respect to the distance between the guessed and the observation data is equivalent to the global MSE minimization.

[1] L. G. Alexopoulos, J. Saez-Rodriguez, B. Cosgrove, D. A. Lauffenburger, P. Sorger, Networks inferred from biochemical data reveal profound differences in toll-like receptor and inflammatory signaling between normal and transformed hepatocytes, Molecular & Cellular Proteomics 9 (9) (2010) 1849–1865.

[2] S. Kauffman, Metabolic stability and epigenesis in randomly constructed genetic nets, Journal of Theoretical Biology 22 (3) (1969) 437–467.

[3] R. Wang, A. Saadatpour, R. Albert, Boolean modeling in systems biology: an overview of methodology and applications, Phys Biol 9 (5).

[4] N. Berestovsky, L. Nakhleh, An evaluation of methods for inferring boolean networks from time-series data, PLoS ONE 8 (6) (2013) e66031.

[5] A. MacNamara, C. Terfve, D. Henriques, B. Bernabe, J. Saez-Rodriguez, State-time spectrum of signal transduction logic models, Phys Biol 9 (4).

[6] M. Bansal, V. Belcastro, A. Ambesi-Impiombato, D. di Bernardo, How to infer gene networks from expression profiles, Molecular Systems Biology 3 (2007) 78.

[7] R. D. Smet, K. Marchal, Advantages and limitations of current network inference methods, Nat. Rev. Microbiol. 8 (10) (2010) 717–729.

[8] S. Maetschke, P. Madhamshettiwar, M. J. Davis, M. Ragan, Supervised, semi-supervised and unsupervised inference of gene regulatory networks, Brief. Bioinformatics 15 (2) (2014) 195–211.

[9] H. Busch, D. Camacho-Trullio, Z. Rogon, K. Breuhahn, P. Angel, R. Eils, A. Szabowski, Gene network dynamics controlling keratinocyte migration, Molecular Systems Miology 4 (1) (2008) 199.

[10] R. Porreca, E. Cinquemani, J. Lygerosn, G. Ferrari-Trecate, Identification of genetic network dynamics with unate structure, Bioinformatics 26 (9) (2010) 1239–1245.

[11] S. M. Hill, L. M. Heiser, T. Cokelaer, M. Unger, N. K. Nesser, D. E. Carlin, Y. Zhang, A. Sokolov, E. O. Paull, C. K. Wong, K. Graim, A. Bivol, H. Wang, F. Zhu, et. al, Inferring causal molecular networks: empirical assessment through a community-based effort, Nat. Methods 13 (4) (2016) 310–318.

[12] M. Ostrowski, L. Paulevé, T. Schaub, A. Siegel, C. Guziolowski, Boolean Network Identification from Multiplex Time Series Data, in: O. Roux, J. Bourdon (Eds.), CMSB 2015 - 13th conference on Computational Methods for Systems Biology, Vol. 9308 of Lecture Notes in Computer Science, Springer International Publishing, Nantes, France, 2015, pp. 170–181.

[13] J. Aracena, E. Goles, A. Moreira, L. Salinas, On the robustness of update schedules in boolean networks, Biosystems 97 (1) (2009) 1–8.

[14] A. Cheng, J. Esparza, J. Palsberg, Complexity results for 1-safe nets, Theor. Comput. Sci. 147 (1&2) (1995) 117–136.

[15] C. Baral, Knowledge Representation, Reasoning and Declarative Problem Solving, Cambridge University Press, 2003.

[16] M. Gebser, R. Kaminski, B. Kaufmann, T. Schaub, Answer Set Solving in Practice, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan and Claypool Publishers, 2012.

[17] M. Gebser, B. Kaufmann, T. Schaub, Multi-threaded ASP solving with clasp, Theory and Practice of Logic Programming 12 (4-5) (2012) 525–545.

[18] M. Gebser, B. Kaufmann, R. Otero, J. Romero, T. Schaub, P. Wanko, Domain-specific heuristics in answer set programming, in: Proceedings of the 27th National Conference on Artificial Intelligence (AAAI'13), AAAI Press, 2013, pp. 350–356.

[19] E. M. Clarke, E. A. Emerson, Design and synthesis of synchronization skeletons using branching-time temporal logic, in: Logic of Programs, Springer-Verlag, London, UK, 1981, pp. 52–71.

[20] J. Krumsiek, S. Polsterl, D. M. Wittmann, F. J. Theis, Odefy–from discrete to continuous models, BMC Bioinformatics 11 (2010) 233.

[21] S. Klamt, J. Saez-Rodriguez, J. A. Lindquist, L. Simeoni, E. Gilles, A methodology for the structural and functional analysis of signaling and regulatory networks, BMC Bioinformatics 7 (2006) 56.

[22] O. Sahin, H. Frohlich, C. Lobke, U. Korf, S. Burmester, M. Majety, J. Mattern, I. Schupp, C. Chaouiya, D. Thieffry, A. Poustka, S. Wiemann, T. Beissbarth, D. Arlt, Modeling ERBB receptor-regulated G1/S transition to find novel targets for de novo trastuzumab resistance, BMC Systems Biology 3 (2009) 1.

[2]For computing subset minimal solutions, we use a special enumeration algorithm described in [18]

[23] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, A. Tacchella, NuSMV 2: An opensource tool for symbolic model checking, in: Computer Aided Verification, Vol. 2404 of LNCS, Springer Berlin / Heidelberg, 2002, pp. 241–268.
[24] A. Biere, A. Cimatti, E. Clarke, O. Strichman, Y. Zhu, Bounded model checking, Advances in Computers 58.
[25] L. Paulevé, Goal-Oriented Reduction of Automata Networks, in: CMSB 2016 - 14th conference on Computational Methods for Systems Biology, Vol. 9859 of Lecture Notes in Bioinformatics, Springer, 2016.
[26] C. Guziolowski, S. Videla, F. Eduati, S. Thiele, T. Cokelaer, A. Siegel, J. Saez-Rodriguez, Exhaustively characterizing feasible logic models of a signaling network using answer set programming, Bioinformatics 29 (18) (2013) 2320–2326.
[27] J. Saez-Rodriguez, L. G. Alexopoulos, J. Epperlein, R. Samaga, D. Lauffenburger, S. Klamt, P. Sorger, Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction, Molecular Systems Biology 5 (331).
[28] H. Klarner, A. Streck, D. Šafránek, J. Kolčák, H. Siebert, Parameter identification and model ranking of thomas networks, in: Computational Methods in Systems Biology, Springer Berlin / Heidelberg, 2012, pp. 207–226.
[29] E. Gallet, M. Manceny, P. L. Gall, P. Ballarini, An ltl model checking approach for biological parameter inference, in: Formal Methods and Software Engineering, Vol. 8829 of LNCS, Springer Berlin / Heidelberg, 2014, pp. 155–170.
[30] M. Gebser, A. Harrison, R. Kaminski, V. Lifschitz, T. Schaub, Abstract Gringo, Theory and Practice of Logic Programming 15 (4-5) (2015) 449–463, available at `http://arxiv.org/abs/1507.06576`.