Reduction Techniques for Boolean Networks

Argyris, Georgios

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

Technical University of Denmark
Department of Applied Mathematics and Computer Science

Reduction Techniques for Boolean Networks

Supervisors
Alberto Lluch Lafuente
Andrea Vandin

Student
Georgios Argyris

January 30, 2023

Abstract

Boolean Networks (BNs) are popular qualitative formalisms for the modelling of biological systems. However, their analysis suffer from the state space explosion; there are exponentially many states on the number of BN variables. To enhance tractable analysis of the underlying systems, we facilitate scientists and engineers with techniques and tools. This thesis is a compendium of four research articles that introduce two reduction techniques and their software implementation.

The thesis is separated into two parts. The first part gives a concise, semiformal, and meaningful overview of our research work. Here we navigate the reader through the second part of the thesis, the Appendix, which includes our contributions in detail. The first paper introduces Boolean backward equivalence, the second article redesigns Boolean backward equivalence method and extends the applications, the third introduces generalised forward bisimulation for the reduction of general dynamical systems (also Boolean Networks), and the fourth article presents the software implementation of Boolean backward equivalence.

Keywords— Boolean Networks, Reduction

Abstract

Boolean Netværk (BN) er populære qualitative formalismer for modelling af biologiske systemer. Imidlertid lider deres analyse af tilstandseksplosion dvs. der er exponentielt mange tilstande i antallet af BN variabler. For at forbedre gennemførlig analyse af underliggende systemer, skal forskere faciliteres med teknikker og værktøjer. Denne afhandling introducerer to reduktionsteknikker og deres softwareimplementering.

Afhandlingen består af to dele. Den første del giver et koncist, semiformelt og meningsfuldt overblik af vores forskningsarbejde. Her navigerer vi læseren gennem den anden del af afhandlingen, bilaget, der omfatter vores bidrag i detaljer. Den første artikel introducerer Boolean backward equivalence, den anden artikel redesigner Boolean backward Equivalence metode og udvider applikationerne, den tredje introducerer generaliseret forward bisimulation for reduktion af generelle dynamiske systemer (også Boolean netværk), og den fjerde artikel præsenterer softwareimplementeringen af Boolean backward equivalence.

Keywords— Boolean Netværk, Reduktion

Abstract

Τα δίκτυα Μπουλ είναι δημοφιλείς ποιοτικοί φορμαλισμοί για την μοντελοποίηση βιολογικών συστημάτων. Ωστώσο, υποφέρουν από έκρηξη του χώρου καταστάσεων· δηλαδή ο αριθμός των καταστάσεων είναι εκθετικός ως προς τον αριθμό των μεταβλητών. Με αυτή την εργασία διευκολύνουμε επιστήμονες και μηχανικούς με τεχνικές και εργαλεία που τιθασεύουν την ανάλυση των συστημάτων. Αυτό το σύγγραμα συνοδεύει τέσσερα ερευνητικά άρθρα που εισάγουν δύο νέες τεχνικές μείωσης και την υλοποίηση τους σε λογισμικό.

Το σύγγραμα χωρίζεται σε δύο μέρη. Το πρώτο δίνει μία συνοπτική, ημιτυπική, αλλά ουσιαστική επισκόπηση της ερευνετικής μας δραστηριότητας. Σε αυτό το μέρος, καθοδηγούμε τον αναγνώστη στο δεύτερο μέρος του συγγράματος, το παράρτημα, που εμπεριέχει τις συμβολές μας λεπτομερειακά. Το πρώτο άρθρο εισάγει την Boolean Backward Equivalence, το δεύτερο την ανασχεδιάζει και επεκτείνει τις εφαρμογές, το τρίτο παρουσιάζει την γενικευμένη εμπρόσθια διπροσομοίωση (Generalised Forward Bisimulation) για την μείωση γενικευμένων δυναμικών συστημάτων (επομένως και δικτύων Μπουλ), και το τέταρτο παρουσιάζει κάποιες πτυχές της προγραμματιστικής υλοποίησης της Boolean Backward Equivalence.

Keywords— Δίκτυα Μπουλ, μείωση διαστασιμότητας

Acknowledgements

This thesis is a result of 3 years collaboration with Alberto Lluch Lafuente, Andrea Vandin, Max Tschaikowski and Mirco Tribastone. Thanks a lot to them for the nice collaboration.[1] Special thanks to the Technical University of Denmark that provided a nice environment for the execution of the research project, and the DFF (project REDUCTO 9040-00224B) that funded my studies.

Last but no least, thanks to the the institutions that hosted me during my research: the IMT School for Advanced Studies in Lucca (System Modelling and Analysis group), the Goethe University in Frankfurt (Molecular Bioinformatics group), and the University of Bordeaux (group of Modelling and Technologies for Verification).

The Ph.D. program has been completed at the Technical University of Denmark, Department of Applied Mathematics and Computer Science, in the Section of Software System Engineering. Alberto Lluch Lafuente is employed by the Technical University of Denmark and Andrea Vandin is employed by the Sant'Anna School of Advanced studies in Pisa and the Technical University of Denmark.

[1] The writer of the manuscript is responsible for the way that the material is presented in the thesis.

Guidelines for the reader

The reader shall be familiar with logic, validity of logical formulas, equivalence relations, partitions, bisimulations, isomorphisms, and monoids.

The thesis is organized as follows: in Section 1 (Preliminaries) we provide the basic notions, in Section 2 we highlight the importance of reduction with a case study, in Section 3 (Contributions) we present the two reduction methods, in Section 4 (Related work) we focus on relevant work on the reduction of Boolean networks, and in Section 5 we motivate the reader to new horizons for further investigations.

Contents

1 Preliminaries

Qualitative modeling provides a useful framework for modeling biological processes and signalling pathways. Popular models are *Boolean Networks* (BNs) [1]: discrete-time dynamical systems with variables taking values in the Boolean domain ($\{0,1\}$). Each variable is assigned an update function, defined by logical rules, that governs the value of the variable (see left part of Fig. 1). Formally, a BN is defined as follows:

$$
\begin{aligned}
x_1(t+1) &= x_1(t) \wedge \neg x_4(t) \\
x_2(t+1) &= x_1(t) \wedge x_2(t) \\
x_3(t+1) &= \neg x_4(t) \wedge x_1(t) \wedge x_3(t) \\
x_4(t+1) &= x_2(t) \wedge x_3(t) \vee \neg x_1(t)
\end{aligned}
$$

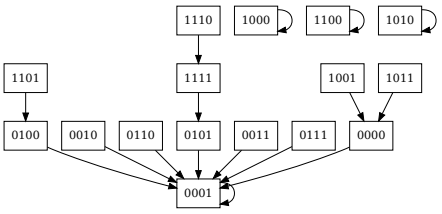


Figure 1: (Left) A discrete-time BN on four variables. (Right) The synchronous STG.

Definition 1. *A BN is a pair (X, F) where $X = \{x_1, ..., x_n\}$ is a set of variables and $F = \{f_{x_1}, ..., f_{x_n}\}$ is a set of update functions, with $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$ being the update function of variable x_i.*

In the example of Fig. 1, we have $X = \{x_1, x_2, x_3, x_4\}$, and $F = \{f_{x_1}, f_{x_2}, f_{x_3}, f_{x_4}\}$ with $f_{x_1} = x_1 \wedge \neg x_4$, $f_{x_2} = x_1 \wedge x_2$, etc. More complex qualitative models are Multi-valued Networks [2] wherein variables take values in a discrete but finite integer domain, e.g. $\{0, 1, 2\}$ (for instance, see *Multi-valued Network case study* of Section 7 in Appendix C). Although Definition 1 will guide the reader through the theoretical development, we will use the formalism of Fig. 1 (left) to explain our concepts in the examples for the sake of simplicity and demonstration.

The state space and the dynamics of a BN are encoded in the state transition graph (STG), which we display in the right part of Fig. 1. The nodes correspond to the states containing an evaluation of the variables, whereas the arrows correspond to the transitions. For example, if the variables x_1, x_2, x_3, x_4 have the value given by the vector of values $(1, 1, 0, 1)$ -upper left box of the STG of Fig. 1-, the simultaneous application of the update functions will map this state to the vector $(0, 1, 0, 0)$. We see that the BNs suffer from the infamous state space explosion since the state space is exponentially large on the number of variables: *if the BN has n variables, the STG has 2^n states.* The contributions of this thesis are clear; we attack the state space explosion by introducing two novel **model-to-model** reduction techniques [3, 4] implemented in ERODE software [5, 6]. Notice that all variables of the BN are updated concurrently, so each state has just one outgoing transition (successor state) in the STG.

Different schemes of synchronization give rise to different dynamics, i.e., different transition set. For example, the fully asynchronous scheme, wherein only one variable is non-deterministically selected and updated, gives rise to a different STG (left part of Fig. 2). A generalization of these schemes is a hybrid synchronization scheme, wherein some subsets of variables are updated at the same time. These subsets form the partition of synchronization (denoted with $\mathcal{K}$) which consists of blocks of variables such that, variables of the same block, are updated concurrently. Note that: in the

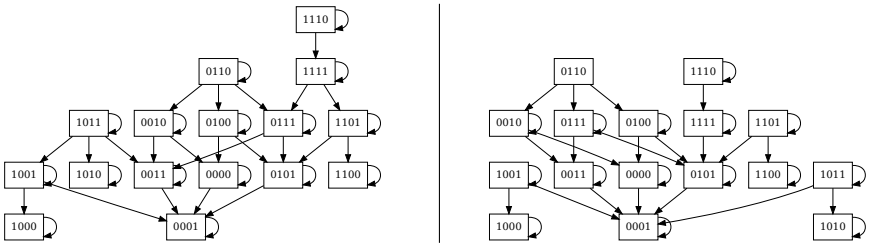


Figure 2: (Left) The fully asynchronous STG wherein at each transition only one variable is non-deterministically selected and updated. (Right) The non-deterministic STG according to the partition of synchronization $\mathcal{K} = \{\{x_1, x_3\}, \{x_2\}, \{x_4\}\}$.

fully asynchronous scheme all variables belong to one (unique) block of the partition ($|\mathcal{K}| = 1$), while in the fully asynchronous scheme each variable belongs to a singleton block ($|\mathcal{K}| = |X|$). For the partition of synchronization one can distinguish two different cases: if we specify an order for the update of the blocks, we obtain an asynchronous yet deterministic STG, while if the blocks are selected non-deterministically for updating, we obtain an asynchronous and non-deterministic STG (see right part of Fig. 2).

Dynamical Properties of Interest. The state space is finite but the variables can be updated infinitely many times. Consequently, the BN will eventually visit a state that has already visited before. Considering the STG of Fig. 2, if the variables x_1, x_2, x_3, x_4 have the values $(1, 0, 0, 0)$ -bottom left box of the STG of Fig. 2- then the update functions will not change these values. These states (like $(1, 0, 0, 0)$) towards which the system tends to evolve and remain are called *attractor* states and their interpretation is crucial in systems biology. In cell differentiation processes, for instance, different attractors correspond to different cell types [7].

An important feature of attractors is the *length* which is typically the number of states that the attractor contains. A steady state is an attractor of length *1*. In the STG of Fig. 2, we see *4* steady states: $(1, 0, 0, 0)$, $(0, 0, 0, 1)$, $(1, 1, 0, 0)$, and $(1, 0, 1, 0)$. If an attractor is not a steady state, it is called a *cyclic* attractor (for a cyclic attractor of length *2*, we refer the reader to the introductory example of Section *Introduction* of Appendix A). In the case of synchronous dynamics, the length of an attractor is also called *period* and the periodic behaviour of cyclic attractors has been utilized to model the cell cycle [8] and the circadian clock [9]. In the case of asynchronous dynamics, the STG may have cycles which are not attractors [10].

Reachability of a BN is another interesting property. This is the ability of a BN model to replicate behaviours observed in time series data [11]: if none of the states matching an observation at a given time is reachable from any state matching an experimental observation at an earlier time, then the BN fails to properly model the system.

2 Reduction is essential

Boolean Networks have been ideal for the modeling of gene regulatory networks (*active/inactive* behaviour of genes), but it is estimated that humans have between *20000* and *25000* genes. From a theoretical viewpoint, the decision of reachability is known to be PSPACE-complete from the logical representation of the update function f [11]. The same holds for the problem of deciding a state belongs to an attractor. This gives a strong argument that the dynamical properties of interest are likely not tractable on models with thousands of variables. To this end, biologists need to be supported with techniques and tools that ease system analysis, and help them with decision making in heterogeneous medical environments, with finding new medical treatments, etc [12].

Formal model reduction ensures fast, low-cost and tractable model analysis while providing formal guarantees for the preserved properties. In the paperwork of the Appendix, we demonstrate that indeed reduction is crucial for the speed-up of computations, and for performing tasks which are otherwise intractable because of our limited computational resources. For relevant results on *STG generation* and *attractor computation*, we refer the reader to Section *Results* of Appendix B wherein we validate the efficiency of our method in the whole GINsim [13] and Biomodels [14] repositories which consist of about 100 qualitative models (both Boolean and Multi-valued Networks [2]). Reduction though has been also utilized to facilitate model checking [12] and model simulation [15].

To motivate the reader, we present an application that complements the large set of experiments done in our papers on a BN with *321* variables. The results are summarized in Table 1. We highlight that the attractors of the original BN cannot be computed with the most efficient tool for attractor identification in synchronous BNs [16]. We display this in first row of Table 1 where *Time-out* means that the computation takes more than 12 hours. Our hypothesis is that we can compute some attractors after reduction. Our proposed techniques behave greedy; by holding no restrictions to our reduction algorithm, we obtain the *Maximal* reduction where we can identify only two attractors (last row of Table 1). Thereby we sacrifice the precision of the model for the sake of reduction performance. Despite greedy, our reduction procedures can be tuned to sacrifice performance for precision; by restricting the reduction power we obtain a different reduced BN with *137* variables, where we can identify a vast amount of attractors.

Model	*Size*	*Attractors*	*Analysis (s)*	*Reduction (s)*
Original	321	*—Time Out—*		-
Restricted reduction	137	8960	175,69	3,071
Maximal	8	2	0,032	0,277

Table 1: Size: number of BN variables. Attractors: Number of attractors (these can be either steady states or cyclic attractors). Analysis (s): attractors computation time in seconds. Reduction (s): Reduction time

More information about the reduction of this model can be found in the Supplementary Material of D. In *C.1, C.2* of Appendix A, we provide empirical reductions that preserve all attractors. These intuitions are based on an abstract graph representation of a BN called *interaction graph* and a relaxed version of Thomas' conjecture [17] relevant to feedback loops in this abstract graph representation. Our approach is similar:

we reduce without corrupting none of the feedback loops.

3 Contributions

Our contributions consist of two novel reduction techniques and their implementation in ERODE software. Articles A, B of the Appendix refer to Boolean Backward Equivalence (BBE) while article C refers to Generalized Forward Bisimulation (GFB). Both techniques are implemented in ERODE. Article D documents the tool support for BBE and provides guidelines for the use of ERODE. Particularly:

- Paper A: Reducing Boolean Networks with Backward Boolean Equivalence. Georgios Argyris, Alberto Lluch Lafuente, Mirco Tribastone, Max Tschaikowski and Andrea Vandin, Conference paper CMSB 2021: Computational Methods in Systems Biology pp 1–18.

  In this paper, we present BBE for synchronous BNs. BBE reduction method identifies disjoint subsets (blocks) of variables that, if they are initialized equally, they remain equal at all time steps. We evaluate the reduction power of our method to the whole GINsim repository, demonstrate that BBE offers speed-ups for attractor computation, and apply BBE to three special case studies.

- Paper B: Reducing Boolean Networks with Backward Boolean Equivalence. Georgios Argyris, Alberto Lluch Lafuente, Mirco Tribastone, Max Tschaikowski and Andrea Vandin, BMC Bioinformatics (Accepted, to appear).

  Here we extend our method to non-deterministic BNs whose variables are updated according to some partition of synchronization. We present new case studies and demonstrate with state-of-the-art tools that BBE reduction renders analysis feasible in the reduced BN while in the original BN the same kind of analysis is infeasible.

- Paper C: Minimization of Dynamical Systems over Monoids. Georgios Argyris, Alberto Lluch Lafuente, Alexander Leguizamon Robayo, Mirco Tribastone, Max Tschaikowski and Andrea Vandin (under preparation).

  This article presents GFB; a reduction method that can be used for reduction of general dynamical systems (DS) like equations of difference, systems of differential equations, BNs, multi-valued networks etc. GFB reduction method identifies blocks of variables such that the DS can be rewritten in terms of the variables of the block and up to a user specified operation. We discuss this method in more detail in Section 3.2.

- Paper D: An Extension of ERODE to Reduce Boolean Networks By Backward Boolean Equivalence. Georgios Argyris, Alberto Lluch Lafuente, Mirco Tribastone, Max Tschaikowski and Andrea Vandin, CMSB 2022: Computational Methods in Systems Biology pp 294–301.

  Here we present the implementation of BBE with ERODE, and some importing and exporting capabilities between different BN formats. ERODE provides also features for BN segmentation as we discuss in Section 3.3.

In the rest of the section, we explain the two complementary methods based on the introductory example of Fig. 1. However, the methods have been applied in detail to overall 10 real world case studies from models found in literature (these studies can be found in the papers of the Appendix). We present BBE in Section 3.1, GFB in Section 3.2, and we discuss some implementation details in Section 3.3.

3.1 Boolean Backward Equivalence

In this section we present the BBE reduction method. We first describe the notion of *BBE partition*, then how to obtain the *BBE reduced BN* up to a BBE partition and, finally, we discuss the *preserved properties of interest*.

BBE partition. BBE is an equivalence relation over the set of variables which induces a partition of the variable set. Each block of a BBE partition satisfies the following property: *if the variables are equal in each block of the partition*, then *the update functions of these variables are equal*. We encode this property as a logical formula whose validity can be checked with a SAT-solver [18]. For the rest of the section, we refer to $\mathcal{P}$ to denote a BBE partition and P to refer to a block of the partition.

Definition 2. *Let BN $B = (X, F)$. A partition $\mathcal{P}$ is a BBE partition if the following formula is valid:*

$$\left(\bigwedge_{\substack{P \in \mathcal{P} \\ x_i, x_j \in P}} \left(x_i = x_j \right) \right) \longrightarrow \bigwedge_{\substack{P \in \mathcal{P} \\ x_i, x_j \in P}} \left(f_{x_i} = f_{x_j} \right) \tag{1}$$

In other words, the previous formula states that if the variables of the block obtain the same value -or are initialized equally-, they always retain the same value. We exemplify the previous definition in the running example of Fig. 1.

Example 3.1. *Consider the BN of Fig. 1*

$$\begin{aligned} x_1(t+1) &= x_1(t) \wedge \neg x_4(t) \\ x_2(t+1) &= x_1(t) \wedge x_2(t) \\ x_3(t+1) &= \neg x_4(t) \wedge x_1(t) \wedge x_3(t) \\ x_4(t+1) &= x_2(t) \wedge x_3(t) \vee \neg x_1(t) \end{aligned}$$

If we set $x_1(t) = x_3(t)$, we have that:

$$x_3(t+1) = \neg x_4(t) \wedge x_1(t) \wedge x_3(t) = \neg x_4(t) \wedge x_1(t) \wedge x_1(t) = \neg x_4(t) \wedge x_1(t) = x_1(t+1)$$

We see that if $x_3(t) = x_1(t)$, it holds that $x_3(t+1) = x_1(t+1)$. This means that x_1, x_3 are BBE equivalent and, consequently, the partition $\mathcal{P} = \{\{x_1, x_3\}, \{x_2\}, \{x_4\}\}$ is a BBE partition.

One may easily check that $\mathcal{P} = \{\{x_1, x_3\}, \{x_2, x_4\}\}$ is not a BBE-partition; if we set $x_2(t) = x_4(t)$, it does not always hold that $x_2(t+1) = x_4(t+1)$.

We next proceed to the computation of the reduced BN which is derived by the original after merging the BBE-equivalent variables into one single variable component.

Computation of BBE reduced BN. We denote with $[\alpha/\beta]$ the replacement of each occurrence of α with β. The reduced BN can be automatically derived according to the following definition:

Definition 3. *The reduction of a BN (X, F) up to a BBE partition $\mathcal{P}$ is the BN (X_P, F_P) where $F_P = \{f_{x_P} : P \in \mathcal{P}\}$, with $f_{x_P} = f_{x_k}\{^{x_i}/_{x_{P'}} : \forall P' \in \mathcal{P}, \forall x_i \in P'\}$ for some $x_k \in P$.*

In words, each variable of the reduced BN corresponds to a block of the BBE partition. Then, we obtain each update function by replacing each variable of a specific block with its corresponding block-variable. We exemplify the previous definition in the running example where the BN is reduced up to the BBE partition $\mathcal{P} = \{\{x_1, x_3\}, \{x_2\}, \{x_4\}\}$.

Example 3.2. *The variables x_1, x_3 are collapsed into one single variable component $x_{1,3}$ whose update function will be either the update function of x_1 or the update function of x_3, after replacing each occurrence of x_1 and each occurrence of x_3 with $x_{1,3}$. In our case, we choose the update function of x_1 and, after replacement, we get $x_{1,3}(t+1) = x_{1,3}(t) \wedge \neg x_4(t)$. The update functions of other variables are also obtained by replacing the occurrences of x_1 and x_3 with $x_{1,3}$. Hence, the reduced BN is the following:*

$$\begin{aligned} x_{1,3}(t+1) &= x_{1,3}(t) \wedge \neg x_4(t) \\ x_2(t+1) &= x_{1,3}(t) \wedge x_2(t) \\ x_4(t+1) &= x_2(t) \vee \neg x_{1,3}(t) \end{aligned}$$

To conclude, if the modeler provides a BN and a partition of the set of variables, a logical formula can determine if this is a BBE partition, and the reduced BN can be automatically computed. However, if the partition provided is not a BBE partition, one may wonder what happens: try all different partitions? This possibility is exhaustive and computationally expensive since the actual number of partitions for n variables is given by the Bell number from the following iterative formula: $B_n = \sum_{k=0}^{n-1} \binom{n-1}{k} B_k$. We solve this problem with a partition refinement algorithm that we roughly describe in Section 3.3.

Properties Preserved. Part of the original STG is related with the reduced STG through a reduction isomorphism (*Lemma 2* in Appendix A). Considering the original STG in the left part of Fig. 3, the blue states with all the transitions between them are preserved. Each blue state of the original STG is mapped to a state in the reduced STG (right part of Fig. 3). For instance, the state $(1, 0, 1, 1)$, where x_1, x_3 have the same value is mapped to $(1, 0, 1)$. The state $(1, 0, 1, 1)$ has two outgoing transitions to steady states $((0, 0, 0, 1), (1, 0, 1, 0))$ while the same happens for the state $(1, 0, 1)$ in the reduced STG (towards $(0, 0, 1)$, $(1, 0, 0)$). The reduced STG is essentially the blue part of the original STG after collapsing the 1st and the 3rd digits into one single digit. In all blue states these two digits are equal.

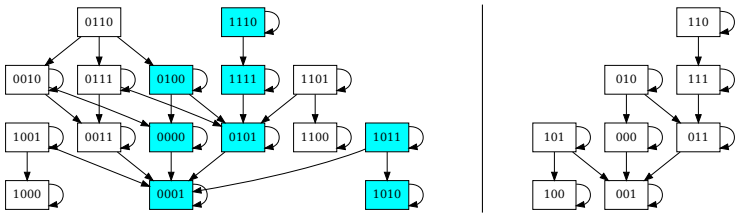


Figure 3: (Left) The STG according to the partition of synchronization $\mathcal{K} = \{\{x_1, x_3\}, \{x_2\}, \{x_4\}\}$ (Right) The BBE-reduced STG according to the partition of synchronization $\mathcal{K} = \{\{x_{1,3}\}, \{x_2\}, \{x_4\}\}$.

We prove that the reduction isomorphism secures the preservation of a special kind of attractors; all attractors wherein the collapsed variables have the same value are preserved -*constant* attractors, *Theorem 3* in Appendix A-. Another immediate consequence of the reduction isomorphism is that the *exact length* of all constant attractors is preserved in the reduced BN. Last but not least, the reduction isomorphism secures the exact number of transitions is preserved between any two states of the reduced STG in the original STG. Consider the state $(0, 1, 0, 0)$ of the original STG which is mapped to $(0, 1, 0)$. State $(0, 1, 0, 0)$ reaches always the same steady state -$(0, 0, 0, 1)$- in two different ways, both containing two transitions. The behaviour of $(0, 1, 0)$ is the same in the reduced STG towards the steady state $(0, 0, 1)$-which corresponds to the state $(0, 0, 0, 1)$-.

We highlight that these properties are preserved in the fully synchronous schema, and in a hybrid scheme considering that the initial partition is (a refinement of) the partition of synchronization $\mathcal{K}$.

3.2 Generalised Forward Bisimulation

In this section we present the GFB reduction method. We first describe a particular form of *GFB partition* called And-FB partition and how one can obtain the And-FB reduced BN. We gradually raise the And-FB definition to GFB definition for the reduction of general dynamical systems and then discuss the preserved properties of interest.

And-FB partition. An And-FB partition is a partition over the set of BN variables. The blocks of an *And-FB* partition satisfy the following property: *the system can be "rewritten" in terms of the conjunction of the variables that belong to the same block.*

Example 3.3. *We first exemplify what "rewritten" means to the BN of Fig. 1:*

$$x_1(t+1) = x_1(t) \wedge \neg x_4(t) \tag{2}$$

$$x_2(t+1) = x_1(t) \wedge x_2(t) \tag{3}$$

$$x_3(t+1) = \neg x_4(t) \wedge x_1(t) \wedge x_3(t) \tag{4}$$

$$x_4(t+1) = x_2(t) \wedge x_3(t) \vee \neg x_1(t) \tag{5}$$

The conjunction by parts of (2) and (3) gives the following:

$$\begin{aligned} x_2(t+1) \wedge x_3(t+1) &= x_1(t) \wedge x_2(t) \wedge \neg x_4(t) \wedge x_1(t) \wedge x_3(t) \\ &= (x_2(t) \wedge x_3(t)) \wedge x_1(t) \wedge \neg x_4(t) \end{aligned}$$

We replace $x_2(t+1) \wedge x_3(t+1)$ with $x_{2,3}(t+1)$, and $x_2(t) \wedge x_3(t)$ with $x_{2,3}(t)$ in all update functions, to obtain the And-FB reduced BN:

$$x_1(t+1) = x_1(t) \wedge \neg x_4(t)$$
$$x_{2,3}(t+1) = x_1(t) \wedge x_{2,3}(t) \wedge \neg x_4(t)$$
$$x_4(t+1) = x_{2,3}(t) \vee \neg x_1(t)$$

The partition $P = \{\{x_1\}, \{x_2, x_3\}, \{x_4\}\}$ *is an And-FB partition.*

The "rewritting" property can be secured by a family of logical formulas whose validity can be checked with a SAT-solver. For the rest of the section, we denote with $\mathcal{P}$ an And-FB partition and with P a block of the partition. As before, we denote with $[\alpha/\beta]$ the replacement of each occurrence of α with β. The definition of an And-FB partition is as follows:

Definition 4. *Let BN* $B = (X, F)$*. A partition* $\mathcal{P}$ *is a And-FB partition if* $\forall P \in \mathcal{P} \wedge \forall x_i, x_j \in P$ *with* $x_i \neq x_j$ *the following formula is valid:*

$$\bigwedge_{P\in\mathcal{P}} \Big(\bigwedge_{x_k\in P} f_{x_k} = \bigwedge_{x_k\in P} f_{x_k}[x_i/1][x_j/(x_i \wedge x_j)] = \bigwedge_{x_k\in P} f_{x_k}[x_j/1][x_i/(x_i \wedge x_j)]\Big)$$

In words, the previous formula secures that for each block of the partition the equality inside the parenthesis holds. The first part of the equality is the conjunction of the update functions of all variables belonging to one specific block. The second part of the equality is the conjunction of the update functions of all variables belonging to this block, after replacing one variable (x_i) with *1*, and the other variable (x_j) with $x_i \wedge x_j$. The third part is the same as the second part after changing the positions of the variables x_i, x_j. *Theorem 3* of Appendix C secures that the validity of the formulas is both sufficient and necessary for us to be able to rewrite the BN.

Computation of the And-FB reduced BN. If the logical formulas of Definition 4 are valid, we can automatically derive the reduced BN according to the following definition.

Definition 5. *The reduction* $B/\mathcal{P}$ *of a BN* $B = (X, F)$*, up to an And-FB* $\mathcal{P}$*, is the BN* $(X_\mathcal{P}, F_\mathcal{P})$ *with* $F_\mathcal{P} = (f_P)_{P\in\mathcal{P}}$ *such that*

$$f_P = \bigwedge_{x_i\in P} f_{x_i}[x_k/1 : x_k \notin X_\mathcal{P}][x_{i_{P'}}/x_{P'} : P' \in \mathcal{P}],$$

where $x_{i_P} \in P$ *is a representative of* $P \in \mathcal{P}$ *and* $X_\mathcal{P} = \{x_{i_P} : P \in \mathcal{P}\}$ *is the set of all representatives.*

We explain how to obtain the reduced BN for the BN of Fig. 1 in the following example.

Example 3.4. *As we explained in Example 3.3, the partition* $P = \{\{x_1\}, \{x_2, x_3\}, \{x_4\}\}$ *is an And-FB partition. The variables* x_2, x_3 *are merged into one single variable component* $x_{2,3}$ *which is the representative of the block* $P = \{x_2, x_3\}$*. Notice that in this case the set of all representatives is the set* $X_\mathcal{P} = \{x_1, x_{2,3}, x_4\}$*.*

The update function of $x_{2,3}$ *is the conjunction of the update functions of the merged variables i.e.:*

$$x_{2,3}(t+1) = x_2(t+1) \wedge x_3(t+1) = (x_2(t) \wedge x_3(t)) \wedge x_1(t) \wedge \neg x_4(t)$$

We get the final form of the update function, by replacing each occurrence of one merged variable with a representative of the block ($x_{2,3}$), and all the other variables with 1. We choose to replace $x_2(t)$ with $x_{2,3}(t)$, and each occurrence of $x_3(t)$ with 1.

The update functions of the other variables (x_1, x_4) are derived similarly: we replace each occurrence of x_2 with $x_{2,3}$, and each occurrence of x_3 with 1. We therefore end up with the reduced BN of Example 3.3.

To sum up, given a BN and a partition of the set of variables, a family of logical formulas can determine if the BN can be rewritten in terms of the $\wedge$ (and) of the variables that belong to the same block of the partition. The reduced BN can be automatically derived as explained in current section. If the formulas do not hold, the partition is plugged to a partition refinement algorithm which splits each block of the partition until it becomes an And-FB partition. The partition refinement can be found in Appendix C and an overview of the automation process is described in Section 3.3.

GFB for arbitrary dynamical systems over monoids. In this section, we raise the definition of GFB to arbitrary dynamical systems. Note that in Definitions 4 and 5 the value *1*, which replaces one of the variables, is the identity element of the monoid $(\mathbb{B}, \wedge)$ i.e. $\forall x \in X : x \wedge 1 = x$. These definitions can be adapted to every operation $\oplus$ as long as the operation forms with the Boolean domain $\mathbb{B} = \{0, 1\}$ a commutative monoid. Hence, the Definition 4, which ensures the "rewritting" property, takes the following form:

Definition 6. *Let BN $B = (X, F)$. A partition $\mathcal{P}$ is a GFB partition if $\forall P \in \mathcal{P} \wedge \forall x_i, x_j \in P$ the following formula is valid:*

$$\bigwedge_{P\in\mathcal{P}} \Big(\bigoplus_{x_k\in P_i} f_{x_k} = \bigoplus_{x_k\in P_i} f_{x_k}[x_i/\mathbf{0}_\oplus][x_j/(x_i \oplus x_j)] = \bigoplus_{x_k\in P_i} f_{x_k}[x_i/(x_i \oplus x_j)][x_j/\mathbf{0}_\oplus] \Big)$$

The orange colour denotes the differences w.r.t. Definition 4. For example, we can adapt to disjunction of variables if we set $\oplus = \vee$ and $\mathbf{0}_\oplus = 0$. Similarly to Definition 5, the reduced DS is defined as follows:

Definition 7. *The reduction $B/\mathcal{P}$ of a BN $B = (X, F)$ up to a GFB $\mathcal{P}$, is the BN $(X_\mathcal{P}, F_\mathcal{P})$ with $F_\mathcal{P} = (f_P)_{P\in\mathcal{P}}$ such that*

$$f_P = \bigoplus_{x_i\in P} f_{x_i}[x_k/1 : x_k \notin X_\mathcal{P}][x_{i_{P'}}/x_{P'} : P' \in \mathcal{P}],$$

where $x_P \in P$ is a representative of $P \in \mathcal{P}$ and $X_\mathcal{P} = \{x_{i_P} : P \in \mathcal{P}\}$ is the set of all representatives.

We can further raise the method to arbitrary discrete-time dynamical systems whose variables take values in a set $\mathbb{M}$ which, when endowed with a modeler specified operation $\oplus$, form a commutative monoid. In summary, GFB supports general discrete-time dynamical systems provided by the following definition:

Definition 8 (Dynamical System)**.** *A discrete-time dynamical system is a pair $D = (X, F)$ where $X = \{x_1, \ldots, x_n\}$ are variables and $F = \{f_{x_1}, \ldots, f_{x_n}\}$ is a set of update functions, where $f_{x_i} : \mathbb{M}^{|X|} \to \mathbb{M}$ is the update function of variable x_i.*

Our generalisation to arbitrary dynamical systems supports equations of difference which can be reduced over the monoid $(\mathbb{R}, \times)$ or the monoid $(\mathbb{R}, +)$. It also encompasses more complex qualitative models called Multi-valued Networks [2] wherein variables take values in a discrete but finite integer domain, e.g. $\{0, 1, 2\}$ (for instance, see *Multi-valued Network case study* of Section 7 in Appendix C). In the latter case, the monoid can be $(\mathbb{Z}_n, min)$ or $(\mathbb{Z}_n, max)$. Next, we give an example of a dynamical system that can be reduced over the monoid $(\mathbb{R}, \times)$.

Example 3.5. *We consider the Lorentz system; a system of ordinary differential equations which is notable for its chaotic solutions for certain parameter values and initial conditions. The equations that describe the evolution are the following:*

$$\begin{aligned} \frac{dx}{dt} &= \sigma(y - x) \\ \frac{dy}{dt} &= x(\rho - z) - y \\ \frac{dz}{dt} &= xy - \beta z \end{aligned} \tag{6}$$

We set $\sigma = 1$, and discretize the system with the Euler method for $\tau = 1$, to obtain the following:

$$\begin{aligned} x(t+1) &= y(t) \\ y(t+1) &= x(t)(\rho + z(t)) \\ z(t+1) &= x(t)y(t) - (\beta - 1)z(t) \end{aligned} \tag{7}$$

Solving for $x(t+1) = x(t)$, $y(t+1) = y(t)$, and $z(t+1) = z(t)$, we find that the original system has 3 critical points: $(0, 0, 0)$, $(\sqrt{\beta(\rho-1)}, \sqrt{\beta(\rho-1)}, \rho - 1)$, and $(-\sqrt{\beta(\rho-1)}, -\sqrt{\beta(\rho-1)}, \rho - 1)$.

Notably, the system can be rewritten up to xy and, hence, we can get a reduction over the monoid $(R, \times)$. The evolution of the system, after collapsing the variables x, y into a single variable component w, is described by the following equations:

$$\begin{aligned} w(t+1) &= w(t)(\rho - z(t)) \\ z(t+1) &= w(t) - (\beta - 1)z(t) \end{aligned} \tag{8}$$

The reduced system has two critical points $(0, 0)$, $(\beta(\rho-1), \rho-1)$. Notice that the critical point $(\beta(\rho-1), \rho-1)$ in the reduced system can be obtained by multiplying the x and y coordinate of the original system's critical points. However, we have to highlight that the partition $\mathcal{P} = \{\{x, y\}, \{z\}\}$ is not a GFB partition since this rewriting must hold $\forall \tau > 0$. For the case of the Lorentz system, this holds only for $\tau = 1$.

Properties preserved. In the left part of Fig. 4, we display the STG of the original BN of Fig. 1, and in the right part of Fig. 4 we display the STG of the reduced BN as we computed it in the *Example 3.3*. States of the original STG with similar colour are mapped to the same state in the reduced STG. Essentially, each state of the reduced STG is derived by the original STG after collapsing the 2nd and the 3rd digit into one single digit according to their $\wedge$. The original STG and the GFB reduced STG are both deterministic and bisimulation equivalent.

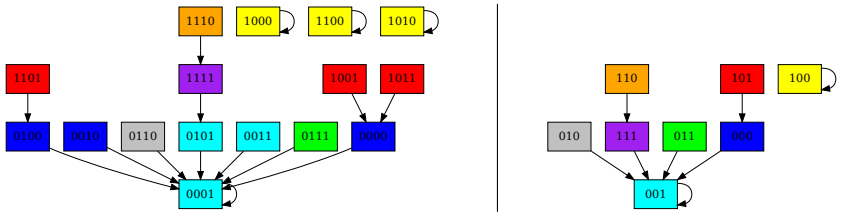


Figure 4: (Left) The original STG in the fully synchronous update scheme. (Right) The GFB-reduced STG.

We prove in Appendix C that several interesting properties are preserved, for instance, attractor states of the original STG are mapped to attractor states of the reduced STG (*Corollary 1*), while reachability between any two states of the same colour in the original STG is preserved in the reduced STG. In Appendix C.1, we present some outstanding cases wherein the computation of the original STG is slow or it cannot be computed, while the reduction speeds-up and renders the computations feasible.

3.3 Automation

One crucial aspect of the proposed reduction methods is the automation; the partition refinement algorithms can identify disjoint sets of variables that satisfy the BBE/GFB criterion or, equivalently, validate the logical formula of Definition 1/Definition 6. The overall procedure is described in Fig. 5.

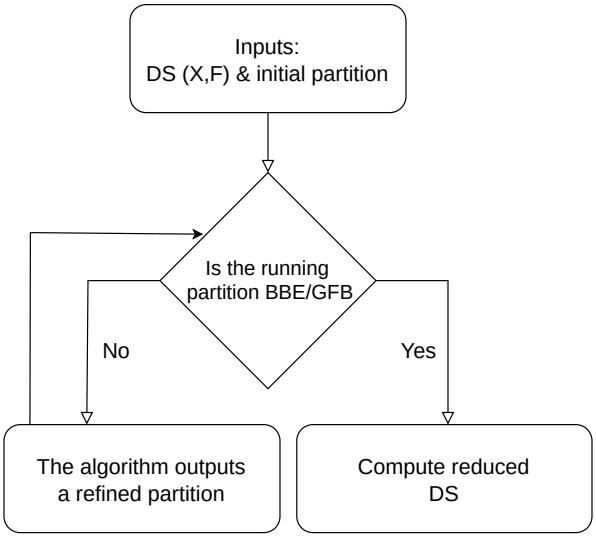


Figure 5: The schematic illustration of the automation.

The inputs of the procedure are a BN/DS (X, F) and an initial partition $\mathcal{P}$ of the set of variables X, which is specified by the modeller (top box of Fig. 5). The procedure

first checks if the partition $\mathcal{P}$ satisfies the BBE/GFB criterion (central rhombus node of Fig. 5). If yes (right arrow of Fig. 5), the running partition $\mathcal{P}$ is a BBE/GFB partition, and the reduced BN/DS is computed (right box of Fig. 5) by merging each block of the BBE/GFB partition into a single variable component. If the modeler provides a partition that is not a BBE/GFB partition (left arrow of Fig. 5), we apply the partition refinement algorithm to refine the partition. The algorithms split each block P and output another partition which is checked if it is a BBE/GFB partition (the feedback arrow from the left box to the central rhombus).

The input of the initial partition provides flexibility to the modeller since, by isolating specific variables of interest in the initial partition, she/he prevents them from merging with other variables. The algorithms output the one unique BBE/GFB partition; for a given BN, every time that we initialize the algorithm with the same initial partition, we end up having the same BBE/GFB partition. (*Theorems* 1 and 2 in Supplementary Material 1 (S1) of Appendix B for the case of BBE, and in *Theorems 1,2* of Appendix C for the case of GFB).

For a closer look to the partition refinement algorithm for the case of BBE, we refer the reader to:

- Appendix A (Section 3.2), where we explain how the algorithm is applied to a BN that models the cortical area development,
- Appendix B where we explain the algorithm to a part of a T-cell model that contains the TRL5 receptor, and
- Appendix D where we explain how the algorithm is applied for the reduction of a neurogenesis BN model.

Software Implementation. In the paper of Appendix D, we present the implementation of BBE with ERODE [2], some importing and exporting capabilities between different BN formats, and a high level overview of the mathematics and the partition refinement algorithm for computing the BBE partition and the reduced model.

ERODE provides special features for BN segmentation. The set of BN variables can be decomposed into three different parts: inputs, internal variables and outputs. Inputs are variables whose update function is either the identity function ($f_x = x$) or a constant ($f_x = 1, f_x = 0$). These variables are inherently backward equivalent so in Appendix A and Appendix B we study different reduction scenarios relevant to them. Outputs are variables whose value does not appear in the update function of any other variable and they are inherently GFB equivalent. Hence, in Appendix C we study different scenarios relevant to output variables. ERODE provides special features to identify these variables and place them either to a unique block or to singleton blocks of the initial partition.

BBE has also been implemented in Colomoto Notebook [19], a framework that combines many tools for interoperability between different system biology tools. An illustration of BBE with Colomoto Notebook is documented [3]. GFB is also supported by ERODE but the implementation is not documented.

[2] `https://www.erode.eu/index.html`

[3] `https://github.com/colomoto/colomoto-docker/blob/for-next/tutorials/ERODE/Reduction%20of%20synchronous%20BNs%20by%20Backward%20Boolean%20Equivalence.ipynb`

4 Related work

The reduction techniques can be classified into two categories according to their domain of reduction: syntactic reduction methods, and semantic reduction methods. Semantic level reductions reduce the STG (i.e. the state space and dynamics of the BN) and, thus, still incurring state space explosion. The methods that we introduced in current thesis reduce the syntax of the BN (i.e. the model). The most popular idea on syntax driven reduction is based on the idea of fast-slow decomposition proposed originally for BNs in [10, 20].

Fast-Slow Decomposition. The main idea is that certain BN variables can get removed by replacing their occurrences in the update functions of other variables with their update functions. Formally, the reduced BN is obtained by the following definition:

Definition 9. *The reduction of B up to a variable x_i is the BN $B' = (X', F')$ where $X' = X \setminus \{x_i\}$ and $F' = \{f_{x_j}[^{x_i}/_{f_{x_i}}] \mid x_j \in X'\}$.*

Next we give an example of the previous definition to our running example.

Example 4.1. *Consider the BN of Fig. 1:*

$$
\begin{aligned}
x_1(t+1) &= x_1(t) \wedge \neg x_4(t) \\
x_2(t+1) &= x_1(t) \wedge x_2(t) \\
x_3(t+1) &= \neg x_4(t) \wedge x_1(t) \wedge x_3(t) \\
x_4(t+1) &= x_2(t) \wedge x_3(t) \vee \neg x_1(t)
\end{aligned}
$$

We remove x_4 from the system, and the occurrences of x_4 in the update functions of other variables will be replaced by the update function of x_4. The reduced BN after simplification is as follows:

$$
\begin{aligned}
x_1(t+1) &= x_1(t) \wedge (\neg x_2(t) \vee \neg x_3(t)) \\
x_2(t+1) &= x_1(t) \wedge x_2(t) \\
x_3(t+1) &= \neg x_1(t) \wedge \neg x_2(t) \wedge x_3(t)
\end{aligned}
$$

We note that this technique has been raised to Multi-valued networks [10] and is implemented in GINsim software [21]. GFB and BBE differ from this method in three aspects: assumptions, limitations, and properties preserved.

Assumptions. In the case of fast-slow decomposition the assumptions are different: the modeller assumes that the absorbed variable (x_4 in the *Example 4.1*) is updated first (or faster) before all the other variables. The assumptions of BBE and GFB are orthogonal; we assume that the aggregated variables are updated at the same time -concurrently-.

Limitations. In the case of fast-slow decomposition, auto-regulated variables cannot be absorbed. These are variables whose state in the next time step depends on its current state, like the variables x_1, x_2 and x_3 in the *Example 4.1.* In all these variables $x_i(t+1)$ depends on $x_i(t)$. The limitations of the methods presented here are different; we cannot always rewrite a BN in terms of any variables up to a modeler specified operation. We explain this in the BN of Fig. 1 where we cannot rewrite the system in terms of $x_1 \vee x_2$ and, thus, x_1, x_2 are not Or-FB equivalent.

Example 4.2. *We try to rewrite the running example of Fig. 1 as the disjunction of x_1, x_2. We have that:*

$$\begin{aligned} x_1(t+1) \vee x_2(t+1) &= x_1(t) \wedge x_2(t) \vee x_1(t) \wedge \neg x_4(t) \\ &= x_1(t) \wedge x_2(t) \wedge x_1(t) \vee x_1(t) \wedge x_2(t) \wedge \neg x_4(t) \\ &= x_1(t) \wedge x_2(t) \vee x_1(t) \wedge x_2(t) \wedge \neg x_4(t) \end{aligned}$$

We observe that we cannot group the expression $x_1(t) \vee x_2(t)$ in order to replace them with $x_{1,2}(t)$. Moreover, the expression $x_1(t) \vee x_2(t)$ does not appear in the update function of x_3, x_4 to be replaced properly. This indicates that the partition $P = \{\{x_1, x_2\}, \{x_3\}, \{x_4\}\}$ is not an Or-FB partition.

Properties preserved. In the case of fast-slow decomposition, steady states are preserved independent of the partition of synchronization as proved in [20]. However, fast-slow decomposition generates spurious behaviours for cyclic attractors; when applied to the synchronous schema cyclic attractors may get shrinked, whereas in the fully asynchronous scheme cyclic attractors may split and form new attractors. Moreover, transient trajectories may also end up being attractors in the reduced BN. In the case of BBE, despite loosing attractors, we secure the exact length of all preserved attractors while in the case of GFB we secure that attractor states of the original BN are mapped to attractor states of the reduced BN.

Reachability is not preserved in the case of fast slow decomposition when the method is used in the fully asynchronous schema [10], while in the case of synchronous schema no relevant reachability results have been published. On the other hand, GFB preserves the reachability and exact number of transitions between any two original states in the reduced STG, concerning that the merged variables are updated concurrently. BBE also preserves all reachability and exact number of transitions between any two original preserved states.

Other relevant work. Other techniques have different limitations: some methods remove *output/leaf* variables [22, 23] (variables that do not appear in the update functions of other variables) or *frozen* ones (variables that stabilize to the same value after some iterations independently of the initial conditions) [24]. Nevertheless, leaf variables usually correspond to the "response" of the modelled system [22, 25], so their removal may not be biologically plausible, while frozen variables can only be identified statistically which is inevitably error prone. We note that the motivation of [24] and [23] are different from ours. Particularly, in [24] the authors perform frozen variable removal in order to justify that complex systems are compressive and reducible while in [23] the authors reduce (with a method called decimation procedure) to investigate the effect of reduction to *stability*. Stability is the ability of the BN to converge to the same attractor when starting from slightly different initial conditions.

5 Future Work and Conlusion

Future Work. The reduction techniques for BNs presented in this thesis have also been implemented for various other formalisms like systems of differential equations [26] and chemical reaction networks [27] but, yet, native implementation to other formalisms (like Petri Nets [28]) is still missing.

BBE has been extended to BNs that are updated according to a hybrid synchronization schema (governed by the partition of synchronization) that leads to a non-deterministic STG. Our intuition is that the same idea can be applied to GFB, which will give raise to bisimulations of non-deterministic STGs. Consider the running example of 1 while its variables are updated according to the partition of synchronization $\mathcal{K} = \{\{x_1, x_4\}, \{x_2, x_3\}\}$. By plugging this initial partition to the partition refinement algorithm of Section 5 of Appendix C, we obtain the GFB partition $\mathcal{P} = \{\{x_1\}, \{x_2, x_3\}, \{x_4\}\}$. The GFB reduced BN is calculated as in the Example 3.4 while its corresponding STG if given in the right part of Fig. 6 according to the partition of synchronization $\mathcal{K} = \{\{x_1, x_4\}, \{x_{2,3}\}\}$. Notice that the original and the reduced STG are non-deterministic yet bisimulation-equivalent.

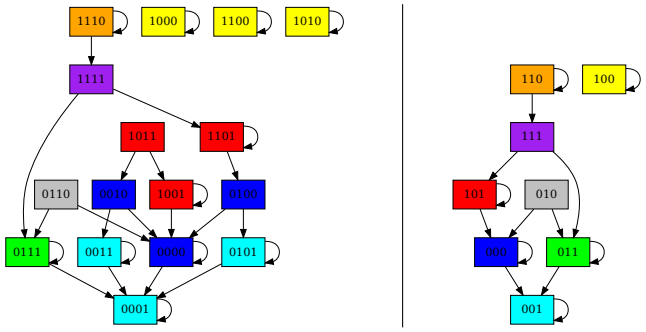


Figure 6: (Left) The original STG according to the partition of synchronization $\mathcal{K} = \{\{x_1, x_4\}, \{x_2, x_3\}\}$. States with the same colour are mapped to the same state in the reduced STG. (Right) The GFB-reduced STG according to the partition of synchronization $\mathcal{K} = \{\{x_1, x_4\}, \{x_{2,3}\}\}$.

Up to now, GFB has been introduced for reduction of arbitrary dynamical systems over monoids. The theory can be extended to discrete-time dynamical systems over arbitrary functions or over other algebraic structures, for instance, rings or semirings. BBE (like GFB) can be trivially extended to arbitrary dynamical systems.

Conclusion. We introduced and implemented two novel reduction methods for DSs. We focused on BNs which are discrete-time, and discrete-space dynamical systems whose variables are updated according to logical rules.

The modeler has to specify a DS and a partition of the set of variables. These are plugged in a partition refinement algorithm which identifies disjoint sets of variables with interesting properties. In the case of BBE, the variables of each set satisfy the following property: *if they are initialized equally, they are always updated equally.*. In the case of GFB, the variables of each set satisfy the following property: *the DS can be rewritten in terms of a modeler-specified operation of the variables that belong to*

the same set. The disjoint sets are merged into single variable components to produce the reduced DS. The overall procedure has been implemented in ERODE.

We have also given formal statements about how the original and the reduced DS are related. The two methods are complementary with other methods found in the literature. As discussed in the related work section, the reduction methods should be applied carefully according to the assumptions of the modeller, the limitations of the DS, and the properties that she/he wishes to preserve. Finally, we shall highlight that reduction is important for system analysis; we can identify properties in the reduced DS which cannot be identified in the original DS due to our limited computational resources.

References

[1] S. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," *Journal of Theoretical Biology*, vol. 22, no. 3, pp. 437 – 467, 1969.

[2] R. Thomas, D. Thieffry, and M. Kaufman, "Dynamical behaviour of biological regulatory networks — i. biological role of feedback loops and practical use of the concept of the loop-characteristic state," *Bulletin of mathematical biology*, vol. 57, no. 2, pp. 247–276, 1995.

[3] G. Argyris, A. Lluch Lafuente, M. Tribastone, M. Tschaikowski, and A. Vandin, "Reducing boolean networks with backward boolean equivalence," in *International Conference on Computational Methods in Systems Biology*. Springer, 2021, pp. 1–18.

[4] ——, "Minimization of dynamical systems over monoids," *arXiv e-prints*, pp. arXiv–2206, 2022.

[5] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin, "Erode: a tool for the evaluation and reduction of ordinary differential equations," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2017, pp. 310–328.

[6] G. Argyris, A. L. Lafuente, M. Tribastone, M. Tschaikowski, and A. Vandin, "An extension of erode to reduce boolean networks by backward boolean equivalence," in *International Conference on Computational Methods in Systems Biology*. Springer, 2022, pp. 294–301.

[7] E. Azpeitia, M. Benítez, I. Vega, C. Villarreal, and E. R. Alvarez-Buylla, "Single-cell and coupled grn models of cell patterning in the arabidopsis thaliana root stem cell niche," *BMC systems biology*, vol. 4, no. 1, pp. 1–19, 2010.

[8] J. Behaegel, J.-P. Comet, G. Bernot, E. Cornillon, and F. Delaunay, "A hybrid model of cell cycle in mammals," *Journal of bioinformatics and computational biology*, vol. 14, no. 01, p. 1640001, 2016.

[9] E. Cornillon, J.-P. Comet, G. Bernot, and G. Enée, "Hybrid gene networks: a new framework and a software environment," *advances in Systems and Synthetic Biology*, 2016.

[10] A. Naldi, E. Remy, D. Thieffry, and C. Chaouiya, "Dynamically consistent reduction of logical regulatory graphs," *Theoretical Computer Science*, vol. 412, no. 21, pp. 2207–2218, 2011.

[11] L. Paulevé, J. Kolčák, T. Chatain, and S. Haar, "Reconciling qualitative, abstract, and scalable modeling of biological networks," *Nature communications*, vol. 11, no. 1, pp. 1–7, 2020.

[12] W. Abou-Jaoudé, P. T. Monteiro, A. Naldi, M. Grandclaudon, V. Soumelis, C. Chaouiya, and D. Thieffry, "Model checking to assess t-helper cell plasticity," *Frontiers in bioengineering and biotechnology*, vol. 2, p. 86, 2015.

[13] A. Naldi, D. Berenguier, A. Fauré, F. Lopez, D. Thieffry, and C. Chaouiya, "Logical modelling of regulatory networks with ginsim 2.3," *Biosystems*, vol. 97, no. 2, pp. 134–139, 2009.

[14] N. Le Novere, B. Bornstein, A. Broicher, M. Courtot, M. Donizelli, H. Dharuri, L. Li, H. Sauro, M. Schilstra, B. Shapiro *et al.*, "Biomodels database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems," *Nucleic acids research*, vol. 34, no. suppl_1, pp. D689–D691, 2006.

[15] L. Grieco, L. Calzone, I. Bernard-Pierrot, F. Radvanyi, B. Kahn-Perles, and D. Thieffry, "Integrative modelling of the influence of mapk network on cancer cell fate decision," *PLoS Comput Biol*, vol. 9, no. 10, p. e1003286, 2013.

[16] E. Dubrova and M. Teslenko, "A sat-based algorithm for finding attractors in synchronous boolean networks," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 8, no. 5, pp. 1393–1399, 2011.

[17] R. Thomas and R. d'Ari, *Biological feedback*. CRC press, 1990.

[18] L. De Moura and N. Bjørner, "Z3: An efficient smt solver," in *International conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2008, pp. 337–340.

[19] A. Naldi, C. Hernandez, N. Levy, G. Stoll, P. T. Monteiro, C. Chaouiya, T. Helikar, A. Zinovyev, L. Calzone, S. Cohen-Boulakia *et al.*, "The colomoto interactive notebook: accessible and reproducible computational analyses for qualitative biological networks," *Frontiers in physiology*, vol. 9, p. 680, 2018.

[20] A. Veliz-Cuba, "Reduction of boolean network models," *Journal of theoretical biology*, vol. 289, pp. 167–172, 2011.

[21] A. Naldi, D. Berenguier, A. Fauré, F. Lopez, D. Thieffry, and C. Chaouiya, "Logical modelling of regulatory networks with ginsim 2.3," *Biosystems*, vol. 97, no. 2, pp. 134–139, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0303264709000665

[22] A. Naldi, P. T. Monteiro, and C. Chaouiya, "Efficient handling of large signalling-regulatory networks by focusing on their core control," in *International Conference on Computational Methods in Systems Biology*. Springer, 2012, pp. 288–306.

[23] S. Bilke and F. Sjunnesson, "Stability of the Kauffman model," *Physical Review E*, vol. 65, no. 1, p. 016129, 2001.

[24] K. A. Richardson, "Simplifying boolean networks," *Advances in Complex Systems*, vol. 8, no. 04, pp. 365–381, 2005.

[25] O. Rodríguez-Jorge, L. A. Kempis-Calanis, W. Abou-Jaoudé, D. Y. Gutiérrez-Reyna, C. Hernandez, O. Ramirez-Pliego, M. Thomas-Chollier, S. Spicuglia, M. A. Santana, and D. Thieffry, "Cooperation between t cell receptor and toll-like receptor 5 signaling for cd4+ t cell activation," *Science signaling*, vol. 12, no. 577, 2019.

[26] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin, "Efficient syntax-driven lumping of differential equations," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2016, pp. 93–111.

[27] ——, "Forward and backward bisimulations for chemical reaction networks," *arXiv preprint arXiv:1507.00163*, 2015.

[28] J. Meseguer and U. Montanari, "Petri nets are monoids," *Information and computation*, vol. 88, no. 2, pp. 105–155, 1990.

[29] H. Klarner, A. Streck, and H. Siebert, "Pyboolnet: a python package for the generation, analysis and visualization of boolean networks," *Bioinformatics*, vol. 33, no. 5, pp. 770–772, 2017.

[30] A. Mbodj, G. Junion, C. Brun, E. E. Furlong, and D. Thieffry, "Logical modelling of drosophila signalling pathways," *Molecular BioSystems*, vol. 9, no. 9, pp. 2248–2258, 2013.

[31] D. P. Cohen, L. Martignetti, S. Robine, E. Barillot, A. Zinovyev, and L. Calzone, "Mathematical modelling of molecular pathways enabling tumour cell invasion and migration," *PLoS Comput Biol*, vol. 11, no. 11, p. e1004571, 2015.

6 Appendix

A PAPER I: Reducing Boolean Networks with Backward Boolean Equivalence

Reproduced with permission from Springer Nature

Reducing Boolean Networks with Backward Boolean Equivalence - extended version*

Georgios Argyris[1 [0000−0002−3203−0410]], Alberto Lluch Lafuente[1 [0000−0001−7405−0818]], Mirco Tribastone[2 [0000−0002−6018−5989]], Max Tschaikowski[3 [0000−0002−6186−8669]], and Andrea Vandin[4,1 [0000−0002−2606−7241]]

[1] DTU Technical University of Denmark, Kongens Lyngby, Denmark
[2] IMT School for Advanced Studies Lucca, Italy
[3] University of Aalborg, Denmark
[4] Sant'Anna School for Advanced Studies, Pisa, Italy

Abstract. Boolean Networks (BNs) are established models to qualitatively describe biological systems. The analysis of BNs might be infeasible for medium to large BNs due to the state-space explosion problem. We propose a novel reduction technique called *Backward Boolean Equivalence* (BBE), which preserves some properties of interest of BNs. In particular, reduced BNs provide a compact representation by grouping variables that, if initialized equally, are always updated equally. The resulting reduced state space is a subset of the original one, restricted to identical initialization of grouped variables. The corresponding trajectories of the original BN can be exactly restored. We show the effectiveness of BBE by performing a large-scale validation on the whole GINsim BN repository. In selected cases, we show how our method enables analyses that would be otherwise intractable. Our method complements, and can be combined with, other reduction methods found in the literature.

Keywords: Boolean Network · State Transition Graph · Attractor Analysis · Exact Reduction · GinSim Repository

1 Introduction

Boolean Networks (BNs) are an established method to model biological systems [28]. A BN consists of Boolean variables (also called nodes) which represent the activation status of the components in the model. The variables are commonly depicted as nodes in a network with directed links which represent influences between them. However, a full descriptive mathematical model underlying a BN consists of a set of Boolean functions, the *update functions*, that govern the Boolean values of the variables. Two BNs are displayed on top of Fig. 1. The BN on the left has three variables x_1, x_2, and x_3, and the BN on

* Partially supported by the DFF project REDUCTO 9040-00224B, the Poul Due Jensen Foundation grant 883901, and the PRIN project SEDUCE 2017TWRCNB.

the right has two variables $x_{1,2}$ and x_3. The dynamics (the state space) of a BN is encoded into a *state transition graph* (STG). The bottom part of Fig. 1 displays the STGs of the corresponding BNs. The boxes of the STG represent the BN *states*, i.e. vectors with one Boolean value per BN variable. A directed edge among two STG states represents the evolution of the system from the source state to the target one. The target state is obtained by synchronously applying all the update functions to the activation values of the source state. There exist BN variants with other update schema, e.g. asynchronous non-deterministic [47] or probabilistic [43]. Here we focus on the synchronous case. BNs where variables are *multivalued*, i.e. can take more than two values to express different levels of activation [46], are supported via the use of *booleanization* techniques [18], at the cost, however, of increasing the number of variables.

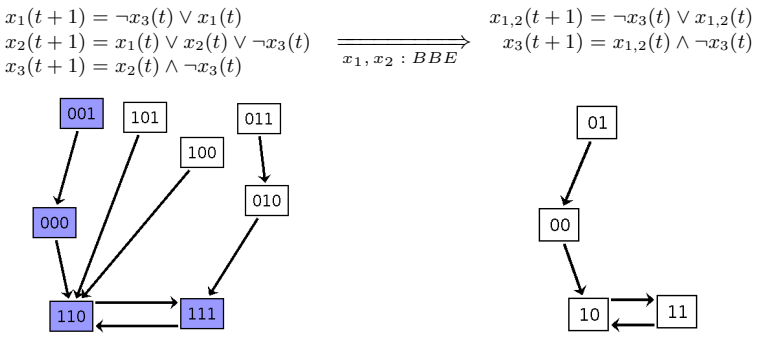


Fig. 1: A BN (top-left), its STG (bottom-left), the BBE-reduced BN (top-right) and its (reduced) STG (bottom-right).

BNs suffer from the state space explosion problem: there are exponentially many STG states with respect to the number of BN variables. This hampers BN analysis in practice, calling for reduction techniques for BNs. There exist manual or semi-automated ones based on domain knowledge. Such empirical reductions have several drawbacks: being semi-automated, they are error-prone, and do not scale. Popular examples are those based on the idea of *variable absorption*, proposed originally in [34,48,41]. The main idea is that certain BN variables can get *absorbed* by the update functions of their target variables by replacing all occurrences of the absorbed variables with their update functions. Other methods automatically remove *leaf* variables (variables with 0 outgoing links) or *frozen* variables (variables that stabilize after some iterations independently of the initial conditions) [39,3]. Several techniques [23,1] focus on reducing the STGs rather than the BN generating them. This requires to construct the original STG, thus still incurring the state space explosion problem.

Our research contributes a novel mathematically grounded method to automatically minimize BNs while exactly preserving behaviors of interest. We present Backward Boolean Equivalence (BBE), which collapses *backward Boolean*

equivalent variables. The main intuition is that two BN variables are BBE-equivalent if they maintain equal value in any state reachable from a state wherein they have the same value. In the STG in Fig. 1 (left), we note that for all states where x_1 and x_2 have same value (purple boxes), the update functions do not distinguish them. Notably, BBE is that it can be checked directly on the BN, without requiring to generate the STG. Indeed, as depicted in the middle of Fig. 1 , x_1 and x_2 can be shown to be BBE-equivalent by inspecting their update functions: If x_1, x_2 have the same value in a state, i.e. $x_1(t) = x_2(t)$, then their update functions will not differentiate them since $x_2(t+1) = x_1(t) \vee x_2(t) \vee \neg x_3(t) = x_1(t) \vee x_1(t) \vee \neg x_3(t) = x_1(t) \vee \neg x_3(t) = x_1(t+1)$. We also present an iterative partition refinement algorithm [36] that computes the largest BBE of a BN. Furthermore, given a BBE, we obtain a *BBE-reduced* BN by collapsing all BBE-equivalent variables into one in the reduced BN. In Fig. 1, we collapsed x_1, x_2 into $x_{1,2}$. The reduced BN faithfully preserves part of the dynamics of the original BN: it exactly preserves all states and paths of the original STG where BBE-equivalent variables have same activation status. Fig. 1 (right) shows the obtained BBE-reduced BN and its STG. We can see that the purple states of the original STG are preserved in the one of the reduced BN.

We implemented BBE in ERODE [10], a freely available tool for reducing biological systems. We built a toolchain that combines ERODE with several tools for the analysis, visualization and reduction of BNs, allowing us to apply BBE to all BNs from the GINsim repository (`http://ginsim.org/models_repository`). BBE led to reduction in 61 out of 85 considered models (70%), facilitating STG generation. For two models, we could obtain the STG of the reduced BN while it is not possible to generate the original STG due to its size. We further demonstrate the effectiveness of BBE in three case studies, focusing on their *asymptotic dynamics* by means of *attractors analysis*. Using BBE, we can identify the attractors of large BNs which would be otherwise intractable.

The article is organized as follows: Section 2 provides the basic definitions and the running example based on which we will explain the key concepts. In Section 3 , we introduce BBE, present the algorithm for the automatic computation of maximal BBEs, and formalize how the STGs of the original and the reduced BN are related. In Section 4, we apply BBE to BNs from the literature. In Section 5 we discuss related works, while Section 6 concludes the paper.

2 Preliminaries

BNs can be represented visually using some graphical representation which, however, might not contain all the information about their dynamics [29]. An example is that of signed interaction (or regulatory) graphs adopted by the tool GinSim [31]. These representations are often paired with a more precise description containing either truth tables [39] or algebraic update functions [45]. In this paper we focus on such precise representation, and in particular on the latter. However, in order to better guide the reader in the case studies, wherein we ma-

$$
\begin{aligned}
x_{Fgf8}(t+1) &= x_{Fgf8}(t) \wedge \neg x_{Emx2}(t) \wedge x_{Sp8}(t) \\
x_{Pax6}(t+1) &= \neg x_{Emx2}(t) \wedge x_{Sp8}(t) \wedge \neg x_{Coup\_tfi}(t) \\
x_{Emx2}(t+1) &= \neg x_{Fgf8}(t) \wedge \neg x_{Pax6}(t) \wedge \neg x_{Sp8}(t) \wedge x_{Coup\_tfi}(t) \\
x_{Sp8}(t+1) &= x_{Fgf8}(t) \wedge \neg x_{Emx2}(t) \\
x_{Coup\_tfi}(t+1) &= \neg x_{Fgf8}(t) \wedge \neg x_{Sp8}(t)
\end{aligned}
$$

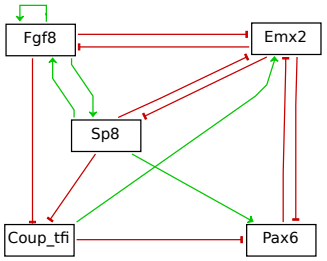


Fig. 2: (Left) the BN of cortical area development from [25]; (Right) its signed interaction graph.

nipulate BNs with a very large number of components, we also introduce signed interaction graphs.

We explain the concepts of current and next sections using the simple BN of Fig. 2 (left) taken from [25]. The model refers to the development of the outer part of the brain: the cerebral cortex. This part of the brain contains different areas with specialised functions. The BN is composed of five variables which represent the gradients that take part in its development: the morphogen *Fgf8* and four transcription factors, i.e., *Emx2, Pax6, Coup_tfi, Sp8*. During development, these genes are expressed in different concentrations across the surface of the cortex forming the different areas.

Fig. 2 (right) displays the signed interaction graph that corresponds to the BN. The green arrows correspond to *activations* whereas the red arrows correspond to *inhibitions*. For example, the green arrow from *Sp8* to *Pax6* denotes that the former promotes the latter because variable x_{Sp8} appears (without negation) in the update function of x_{Pax6}, whereas the red arrow from *Pax6* to *Emx2* denotes that the former inhibits the latter because the negation of x_{Pax6} appears in the update function of x_{Emx2}.

We now give the formal definition of a BN:

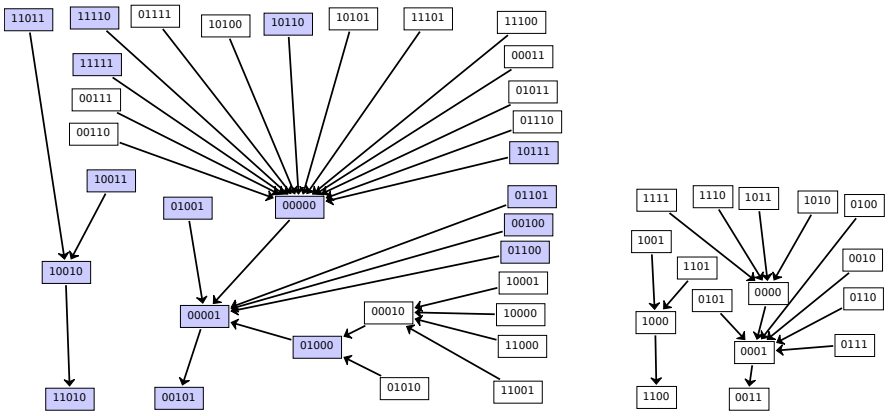
Definition 1. *A BN is a pair (X, F) where $X = \{x_1, ..., x_n\}$ is a set of variables and $F = \{f_{x_1}, ..., f_{x_n}\}$ is a set of update functions, with $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$ being the update function of variable x_i.*

A BN is often denoted as $X(t+1) = F(X, t)$, or just $X = F(X)$. In Fig. 2 we have $X = \{x_{Fgf8}, x_{Pax6}, x_{Emx2}, x_{Sp8}, x_{Coup\_tfi}\}$.

The *state* of a BN is an evaluation of the variables, denoted with the vector of values $\mathbf{s} = (s_{x_1}, \ldots, s_{x_n}) \in \mathbb{B}^n$. The variable x_i has the value s_{x_i}. When the update functions are applied synchronously, we have synchronous transitions between states, i.e. for $\mathbf{s}, \mathbf{t} \in \mathbb{B}^n$ we have $\mathbf{s} \to \mathbf{t}$ if $\mathbf{t} = F(\mathbf{s}) = (f_{x_1}(\mathbf{s}), \ldots, f_{x_n}(\mathbf{s}))$.

Suppose that the activation status of the variables x_{Fgf8}, x_{Emx2}, x_{Pax6}, x_{Sp8}, x_{Coup_tfi} is given by the state $\mathbf{s} = (1, 0, 1, 1, 1)$. After applying the update functions, we have $\mathbf{t} = F(\mathbf{s}) = (0, 0, 0, 0, 0)$.

The state space of a BN, called *State Transition Graph (STG)*, is the set of all possible states and state transitions.



(a) STG of BN in Fig. 2 (b) STG of BN in Fig. 4

Fig. 3: The STGs of the BN of Fig. 2 and of its BBE-reduction in. Fig. 4. We use GINsim's visual representation, where self-loops are implicit in nodes without outgoing edges.

Definition 2. *Let $B = (X, F)$ be a BN. We define the state transition graph of B, denoted with $STG(B)$, as a pair (S, T) with $S \subseteq \mathbb{B}^n$ being a set of vertices labelled with the states of B, and $T = \{\boldsymbol{s} \to \boldsymbol{t} \mid \boldsymbol{s} \in S, \boldsymbol{t} = F(\boldsymbol{s})\}$ a set of directed edges representing the transitions between states of B.*

We often use the notation $\mathbf{s} \to^+ \mathbf{t}$ for the transitive closure of the transition relation. The cardinality of the set of states is 2^n, which illustrates the state space explosion: we have exponentially many states on BN variables. Fig. 3(a) displays the STG of the BN in Fig. 2.

Several BN properties are identified in STGs, e.g. attractors, basins of attraction, and transient trajectories [42]. Attractors are sets of states towards which a system tends to evolve and remain [27]. They are often associated with the interpretation of the underlying system; for example, Kauffman equated attractors with different cell types [20]. Hence, the main reduction methods that have been developed in the literature so far concentrate on how they affect the asymptotic dynamics i.e. the number of attractors and the distribution of their lengths. We define an attractor as follows:

Definition 3. ***(Attractor)*** *Let $B = (X, F)$ be a BN with $STG(B) = (S, T)$. We say that a set of states $A \subseteq S$ is an* attractor *iff*

1. $\forall \boldsymbol{s}, \boldsymbol{s}' \in A$, $\boldsymbol{s} \to^+ \boldsymbol{s}'$, *and*
2. $\forall \boldsymbol{s} \in A, \forall \boldsymbol{s}' \in S$, $\boldsymbol{s} \to^+ \boldsymbol{s}'$ *implies* $\boldsymbol{s}' \in A$.

Attractors are hence just absorbing strongly connected components in the STG. An attractor A such that $|A| = 1$ is called a *steady state* (also named *point attractor*). We also denote with $|A|$ the *length* of attractor A.

3 Backward Boolean Equivalence

Our reduction method is based on the notion of backward equivalence, recast for BNs, which proved to be effective for reducing the dimensionality of ordinary differential equations [9,13] and chemical reaction networks [11,6,8]. Section 3.1 introduces *Backward Boolean Equivalence* (BBE), which is an equivalence relation on the variables of a BN, and use it to obtain a reduced BN. Section 3.2 provides an algorithm which iteratively compute the maximal BBE of a BN. Section 3.3 relates the properties of an original and BBE-reduced BN.

We fix a BN $B = (X, F)$, with $|X| = n$. We use R to denote equivalence relations on X and X_R for the induced partition.

3.1 Backward Boolean Equivalence and BN Reduction

We first introduce the notion of *constant* state on an equivalence relation R.

Definition 4. ***(Constant State)*** *A state $\boldsymbol{s} \in \mathbb{B}^n$ is* constant *on R if and only if $\forall (x_i, x_j) \in R$ it holds that $s_{x_i} = s_{x_j}$.*

Consider our running example and an equivalence relation R given by the partition $X_R = \{\{x_{Sp8}, x_{Fgf8}\}, \{x_{Pax6}\}, \{x_{Emx2}\}, \{x_{Coup\_tfi}\}\}$. The states constant on R are colored in purple in Fig. 3. For example, the state $\mathbf{s} = (1, 0, 1, 1, 1)$ is constant on R because $s_{Sp8} = s_{Fgf8}$ (the first and fourth positions of $\mathbf{s}$, respectively). On the contrary, $(1, 0, 1, 0, 1)$ is not constant on R.

We now define *Backward Boolean Equivalence (BBE).*

Definition 5. ***(Backward Boolean Equivalence)*** *Let $B = (X, F)$ be a BN, X_R a partition of the set X of variables, and $C \in X_R$ a class of the partition. A partition X_R is a* Backward Boolean Equivalence (BBE) *if and only if the following formula is valid:*

$$\Phi^{X_R} \equiv \left(\bigwedge_{\substack{C \in X_R \\ x, x' \in C}} \Big(x = x' \Big) \right) \longrightarrow \bigwedge_{\substack{C \in X_R \\ x, x' \in C}} \Big(f_x(X) = f_{x'}(X) \Big)$$

Φ^{X_R} says that if for all equivalence classes C the variables in C are equal, then the update functions of variables in the same equivalence class stay equal.

In other words, R is a BBE if and only if for all $\mathbf{s} \in \mathbb{B}^n$ constant on R it holds that $F(\mathbf{s})$ is constant on R. BBE is a relation where the update functions F preserve the "constant" property of states. The partition $X_R = \{\{x_{Sp8}, x_{Fgf8}\}, \{x_{Pax6}\}, \{x_{Emx2}\}, \{x_{Coup\_tfi}\}\}$ described above is indeed a BBE. This can be verified on the STG: all purple states (the constant ones) have outgoing transitions only towards purple states.

We now define the notion of BN reduced up to a BBE R. Each variable in the reduced BN represents one equivalence class in R. We denote by $f\{^a/_b\}$ the term arising by replacing each occurrence of b by a in the function f.

Definition 6. *The reduction of B up to R, denoted by B/R, is the BN (X_R, F_R) where $F_R = \{f_{x_C} : C \in X_R\}$, with $f_{x_C} = f_{x_k}\{^{x_{C'}}/_{x_i} : \forall C' \in X_R, \forall x_i \in C'\}$ for some $x_k \in C$.*

The definition above uses one variable per equivalence class, selects the update function of any variable in such class, and replaces all variables in it with a representative one per equivalence class. Fig. 4 shows the reduction of the cortical area development BN. We selected the update function of x_{Sp8} as the update function of the class-variable $x_{\{Fgf8,Sp8\}}$, and replaced every occurrence of x_{Sp8} and x_{Fgf8} with $x_{\{Fgf8,Sp8\}}$. The STG of such reduced BN is given in Fig. 3(b).

$$
\begin{aligned}
x_{\{Fgf8,Sp8\}}(t+1) &= x_{\{Fgf8,Sp8\}}(t) \wedge \neg x_{\{Emx2\}}(t) \\
x_{\{Pax6\}}(t+1) &= \neg x_{\{Emx2\}}(t) \wedge x_{\{Fgf8,Sp8\}}(t) \wedge \neg x_{\{Coup\_tfi\}}(t) \\
x_{\{Emx2\}}(t+1) &= \neg x_{\{Fgf8,Sp8\}}(t) \wedge \neg x_{\{Pax6\}}(t) \wedge \neg x_{\{Fgf8,Sp8\}}(t) \wedge x_{\{Coup\_tfi\}}(t) \\
x_{\{Coup\_tfi\}}(t+1) &= \neg x_{\{Fgf8,Sp8\}}(t) \wedge \neg x_{\{Fgf8,Sp8\}}(t)
\end{aligned}
$$

Fig. 4: The BBE-reducion of the cortical area development network of Fig. 2.

3.2 Computation of the maximal BBE

A crucial aspect of BBE is that it can be checked directly on a BN without requiring the generation of the STG. This is feasible by encoding the logical formula of Definition 5 into a logical SATisfiability problem [2]. A SAT solver has the ability to check the validity of such a logical formula by checking for the unsatisfiability of its negation ($sat(\neg \Phi^{X_R})$). A partition X_R is a BBE if and only if sat($\neg \Phi^{X_R}$) returns "unsatifiable", otherwise a counterexample (a witness) is returned, consisting of variables assignments that falsify Φ^{X_R}. Using counterexamples, it is possible to develop a partition refinement algorithm that computes the largest BBE that refines an initial partition.

The partition refinement algorithm is shown in Algorithm 1. Its input are a BN and an initial partition of its variables X. A *default* initial partition that leads to the maximal reduction consists of one block only, containing all variables. In general, the modeller may specify a different initial partition if some variables should not be merged together, placing them in different blocks. The output of the algorithm is the largest partition that is a BBE and refines the initial one.

We now explain how the algorithm works for input the cortical area development BN and the initial partition $X_R = \{\{x_{Fgf8}, x_{Emx2}, x_{Pax6}, x_{Sp8}, x_{Coup\_tfi}\}\}$.

Iteration 1. The algorithm enters the *while* loop, and the solver checks if Φ^{X_R} is valid. X_R is not a BBE, therefore the algorithm enters the second branch of the *if* statement. The solver gives an example satisfying $\neg \Phi^{X_R}$: $s = (s_{x_{Fgf8}}, s_{x_{Pax6}}, s_{x_{Emx2}}, s_{x_{Sp8}}, s_{x_{Coup\_tfi}}) = (0,0,0,0,0)$. Since $t = F(s) = (0,0,0,0,1)$, the *for* loop partitions G into $X_{R_1} = \{\{x_{Fgf8}, x_{Pax6}, x_{Emx2}\ x_{Sp8}\}, \{x_{Coup\_tfi}\}\}$. The state $t = (0,0,0,0,1)$ is now constant on X_{R_1}.

Algorithm 1: Compute the maximal BBE that refines the initial partition X_R for a BN (X, F)

Result: maximal BBE H that refines X_R
H $\leftarrow X_R$;
while *true* **do**
  if Φ^H *is valid* **then**
    return H ;
  else
    s $\leftarrow$ get a state that satisfy $\neg\Phi^H$;
    $H' \leftarrow \emptyset$;
    for $C \in H$ **do**
      $C_0 = \{x_i \in C : f_{x_i}(\mathbf{s}) = 0\}$;
      $C_1 = \{x_i \in C : f_{x_i}(\mathbf{s}) = 1\}$;
      $H' = H' \cup \{C_1\} \cup \{C_0\}$;
    end
    $H \leftarrow H' \setminus \{\emptyset\}$;
  end
end

Iteration 2. The algorithm checks if $\Phi^{X_{R_1}}$ is valid (i.e. if X_{R_1} is a BBE). X_{R_1} is not a BBE. The algorithm gives a counterexample with $s = (0, 0, 0, 0, 1)$ and $t = F(s) = (0, 0, 1, 0, 1)$. The *for* loop refines X_{R_1} into $X_{R_2} = \{\{x_{Fgf8}, x_{Pax6}\ x_{Sp8}\}, \{x_{Emx2}\}, \{x_{Coup\_tfi}\}\}$. X_{R_2} makes $t = (0, 0, 1, 0, 1)$ constant.

Iteration 3. The algorithm checks if G_2 is a BBE. The formula $\neg\Phi^{X_{R_2}}$ is satisfiable, so G_2 is not a BBE, and the solver provides an example with $s = (1, 1, 0, 1, 1)$ and $F(s) = (1, 0, 0, 1, 0)$. Hence, X_{R_2} is partitioned into $X_{R_3} = \{\{x_{Fgf8}, x_{Sp8}\}, \{x_{Pax6}\}\ \{x_{Emx2}\}, \{x_{Coup\_tfi}\}\}$.

Iteration 4. The SAT solver proves that $\Phi^{X_{R_3}}$ is valid.

The number of iterations needed to reach a BBE depends on the counterexamples that the SAT solver provides. As for all partition-refinement algorithms, it can be easily shown that the number of iterations is bound by the number of variables. Each iteration requires to solve a SAT problem which is known to be NP-complete, however we show in Section 4 that we can easily scale to the largest models present in popular BN repositories.

We first show that given an initial partition there exists exactly one *largest* BBE that refines it. [1]

After that, we prove that Algorithm 1 indeed provides the maximal BBE that refines the initial one.

Theorem 1. *Let $BN = (X, F)$ and X_R a partition. There exists a unique maximal BBE H that refines X_R.*

Theorem 2. *Algorithm 1 computes the maximal BBE partition refining X_R.*

[1] All proofs are given in Appendix A

3.3 Relating Dynamics of Original and Reduced BNs

Given a BN B and a BBE R, $STG(B/R)$ can be seen as the subgraph of $STG(B)$ composed of all states of $STG(B)$ that are constant on R and their transitions. Of course, those states are transformed in $STG(B/R)$ by "collapsing" BBE-equivalent variables in the state representation. This can be seen by comparing the STG of the our running example (left part of Fig. 3) and of its reduction (right part of Fig. 3). The states (and transitions) of the STG of the reduced BN correspond to the purple states of the original STG.

Let B be a BN with n variables, $S \subseteq \mathbb{B}^n$ be the states of its STG, and R a BBE for B. We use $S_{|R}$ to denote the subset of S composed by all and only the states constant on R. With $STG(B)_{|R}$ we denote the subgraph of $STG(B)$ containing $S_{|R}$ and its transitions. Formally $STG(B)_{|R} = (S_{|R}, T_{|R})$, where $T_{|R} = T \cap (S_{|R} \times S_{|R})$.

The following lemma formalizes a fundamental property of $STG(B)_{|R}$, namely that all attractors of B containing states constant on R are preserved in $STG(B)_{|R}$.

Lemma 1. ***(Constant attractors)*** *Let $B(X, F)$ be a BN, R be a BBE, and A an attractor. If $A \cap S_{|R} \neq \emptyset$ then $A \subseteq S_{|R}$.*

We now define the bijective mapping $m_R : S_{|R} \leftrightarrow S_R$ induced by a BBE R, where S_R are the states of $STG(B/R)$, as follows: $m_R(\mathbf{s}) = (v_{C_1}, \ldots, v_{C_{|X/R|}})$ where $v_{C_j} = s_{x_i}$ for some $x_i \in C_j$. In words m_R bijectively maps each state of $STG(B)_{|R}$ to their compact representation in $STG(B/R)$. Indeed, $STG(B)_{|R}$ and $STG(B/R)$ are isomorphic, with m_R defining their (bijective) relation. We can show this through the following lemma.

Lemma 2. ***(Reduction isomorphism)*** *Let $B(X, F)$ be a BN and R be a BBE. Then, it holds*

1. *For all states $\boldsymbol{s} \in S_{|R}$ it holds $F_R(m_R(\boldsymbol{s})) = m_R(F(\boldsymbol{s}))$.*
2. *For all states $\boldsymbol{s} \in S_R$ it holds $F(m_R^{-1}(\boldsymbol{s})) = m_R^{-1}(F_R(\boldsymbol{s}))$.*

The previous Lemma ensures that BBE does not generates spurious trajectories or attractors in the reduced system. We can now state the main result of our approach, namely that the BBE reduction of a BN for a BBE R exactly preserves all attractors that are constant on R up to renaming with m_R.

Theorem 3. ***(Constant attractor preservation)*** *Let $B(X, F)$ be a BN, R a BBE, and A an attractor. If $A \cap S_{|R} \neq \emptyset$ then $m_R(A)$ is an attractor for B/R.*

4 Application to BNs from the Literature

We hereby apply BBE to BNs from the GINsim repository. Section 4.1 validates BBE on all models from the repository, while Section 4.2 studies the runtime speedups brought by BBE on attractor-based analysis of selected case studies,

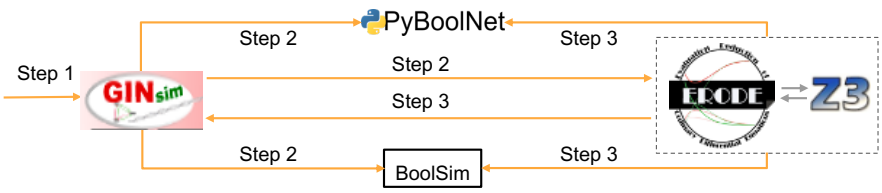


Fig. 5: BBE toolchain. (Step 1) We use GINsim [15] to access its model repository, and (Step 2) export it in the formats of the other tools in the toolchain to perform: STG generation (PyBoolNet [30]), attractor analysis (BoolSim [19]), and BBE reduction (ERODE [10]). (Step 3) We export the reduced models for analysis to PyBoolNet and BoolSim, or to GINsim.

showing cases for which BBE makes the analysis feasible. [2] Section 4.3 compares BBE with the approach based on ODE encoding from [11], showing how such encoding leads to scalability issues and to the loss of reduction power. [3]

The experiments have been made possible by a novel toolchain (Fig. 5) combining tools from the COLOMOTO initiative [33], and the reducer tool ERODE [10] which was extended here to support BBE-reduction. For Algorithm 1 we use the solver Z3 [17] which was already integrated in ERODE.

All experiments were conducted on a common laptop with an Intel Xeon(R) 2.80GHz and 32GB of RAM. We imposed an arbitrary timeout of 24 hours for each task, after which we terminated the analysis. We refer to these cases as *time-out*, while we use *out-of-memory* if a tool terminated with a memory error.

4.1 Large Scale Validation of BBE on BNs

We validate BBE on real-world BNs in terms of the number of BNs that can be reduced and the average reduction ratio.

Configuration. We conducted our investigation on the whole GINsim model repository which contains 85 networks: 29 are Boolean, and 56 are multivalued. In multivalued networks (MNs), some variables have more than 2 activation statuses, e.g. $\{0, 1, 2\}$. These models are automatically *booleanized* [18,14] by GinSim when exporting in the input formats of the other tools in the tool-chain.

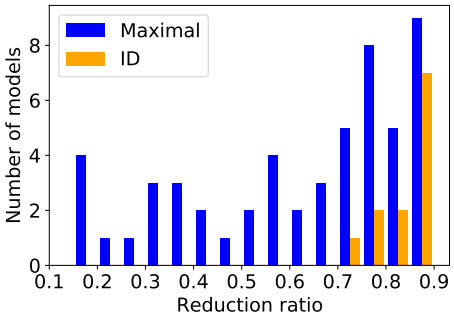
Most of the models in the repository have a specific structure [32] where a few variables are so-called *input variables*. These are variables whose update functions are either a stable function (e.g. $x(t+1) = 0$, $x(t+1) = 1$) or the identity function (e.g. $x(t+1) = x(t)$). These are named 'input' because their values are explicitly set by the modeler to perform experiments campaigns. We investigate two reduction scenarios relevant to input variables. In the first one,

[2] These models are further analysed in Appendix C using initial partitions based on information from the original publications, obtaining better reductions.

[3] Appendix D further studies BBE-induced runtime speedups to STG generation on the repository. We display again cases where BBE makes the analysis feasible.

Algorithm 1 starts with initial partitions that lead to the *maximal reduction*, i.e. consisting of one block only. In the second scenario, we provide initial partitions that isolate inputs in singleton blocks. Therefore, we prevent their aggregation with other variables, and obtain reductions independent of the values of the input variables (we recall that BBE requires related variables to be initialized with same activation value). We call this case *input-distinguished (ID) reduction*.

Results. By using the maximal reduction setting, we obtained reductions on 61 of the 85 models, while we obtained ID reductions on 38 models. We summarize the reductions obtained for the two settings in Fig. 6, displaying the distribution of the reduction ratios $r_m = N_m/N$ and $r_i = N_i/N$, where N, N_m and N_i are the number of variables in the original BN, in the maximal BBE-reduction, and in the ID one, respectively. [4] We also provide the average reduction ratios on the models, showing that it does not substantially change across Boolean or multivalued models. No reduction took more than 3 seconds.



Average reduction ratios

	Maximal	*ID*
BNs	0.66	0.83
MNs	0.68	0.95
ALL	0.67	0.91

Fig. 6: (Left) Distribution of reduction ratios (reduced variables over original ones) on all models from the GINsim repository using the maximal and ID reduction strategy. Each bar counts the number of models with that reduction ratio, starting from 15% up to 90%, with step 5%. (Right) Average reduction ratios for Boolean, Multivalued and all models.

Interpretation. BBE reduced a large number of models (about 72%). In particular, this happened in 24 out of the 29 (83%) Boolean models and in 37 out of 56 (66%) multivalued networks. The average reduction ratio for the maximal and ID strategies are 0.67 and 0.91, respectively. For the former strategy, we get trivial reductions in 22 models wherein only input variables are related. In such trivial cases, the ID strategy does not lead to reduction. In other cases, the target variables of inputs (i.e. variables with incoming edges only from input variables considering the graphical representation of variables) appeared to be

[4] More details can be found in Table 2 of Appendix B.

backward equivalent together with the input variables. This results in reductions with large equivalence classes consisting of input variables and their descendants. These are interesting reductions which get lost using the ID approach, as the input variables get isolated.

4.2 Attractor analysis of selected case studies

Hypothesis. We now investigate the fate of asymptotic dynamics after BBE-reduction, and test the computational efficiency in terms of time needed for attractor identification in the original and reduced models. We expect that BBE-reduction can be utilized to (i) gain fruitful insights into large BN models and (ii) to reduce the time needed for attractor identification.

Configuration. Our analysis focuses on three BNs from the GINsim repository. The first is the Mitogen-Activated Protein Kinases (MAPK) network [26] with 53 variables. The second refers to the survival signaling in large granular lymphocyte leukemia (T-LGL) [52] and contains 60 variables. The third is the merged Boolean model [40] of T-cell and Toll-like receptors (TCR-TLR5) which is the largest BN model in GINsim repository with 128 variables.

Results. The results of our analysis are summarized in Table 1 for the original, ID- and maximal-reduced BN. We present the number of variables (*size*) and of Attractors (*Attr.*), the time for attractor identification on the original model (*An. (s)*) and that for reduction plus attractor identification (*Red. + An. (s)*).

	Original model			*ID reduction*			*Maximal reduction*		
	Size	*Attr.*	*An.(s)*	*Size*	*Attr.*	*Red.+An.(s)*	*Size*	*Attr.*	*Red.+An.(s)*
MAPK Network	53	40	16.50	46	40	15.33	39	17	3.49
T-LGL	60	264	123.43	57	264	86.84	52	6	3.49
TCR-TLR	128	—*Time Out*—		116	—*Time Out*—		95	2	31.29

Table 1: Reduction and attractor analysis on 3 selected case studies.

Interpretation. ID reduction preserves all attractors reachable from any combination of activation values for inputs. This is an immediate consequence of 2, Theorem 3 and the fact that number of attractors in the original and the ID reduced BN is the same (see Table 1). Maximal reduction might discard some attractors. We also note that, despite the limited reduction in terms of obtained number of variables, we have important analysis speed-ups, up to two orders of magnitude. Furthermore, the largest model could not be analyzed, while it took just 30 seconds to analyze its maximal reduction identifying 2 attractors. [5]

[5] There might be further attractors of interest in addition to these. In Appendix C we show how BBE could be used by a modeler by imposing ad-hoc initial partitions to preserve more attractors while reducing more than with the ID stategy.

4.3 Comparison with ODE-based approach from [11]

As discussed, BBE is based on the backward equivalence notion firstly provided for ordinary differential equations (ODEs), chemical reaction networks, and Markov chains [9,11]. Notably, [11] shows how the notion for ODEs can be applied indirectly to BNs via an *odification* technique [49] to encode BNs as ODEs. Such odification transforms each BN variable into an ODE variable that takes values in the continuous interval [0,1]. The obtained ODEs preserve the attractors of the original BN because the equations of the two models coincide when all variables have value either 0 or 1. However, infinitely more states are added for the cases in which the variables do not have integer value.

Scalability. The technique from [11] has been proved able to handle models with millions of variables. Instead, the odification technique is particularly computationally intensive. Due to this, it failed on some models from the GINsim repository, including two from [22], namely *core_engine_budding_yeast_CC* and *coupled_budding_yeast_CC*, consisting of 39 and 50 variables, respectively. Instead, BBE could be applied in less than a second.

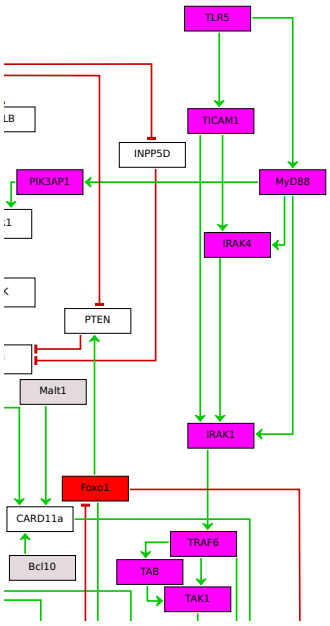


Fig. 7: Excerpt of GINsim's depict of TCR-TLR.

Reduction power. Another example is the *TCR-TLR* model from the previous section. In this case, both the ODE-based and BBE techniques succeeded. However, BBE led to better reductions due to the added non-integer states in the ODEs. Intuitively, the ODE-based technique *counts* incoming influences from equivalence classes of nodes, while BBE only checks whether at least one of such influence is present or not. Figure 7 shows an excerpt of the graphical representation of the model by GINsim. We use background colors of nodes to denote BBE equivalence classes (white denotes singleton classes). We see a large equivalence class of magenta species, 3 of which (*IRAK4*, *IRAK1*, and *TAK1*) receive two influences by magenta species, while the others receive only one. This differentiates the species in the ODE-based technique, keeping only the top four in the *magenta* block, while all the others end up in singleton blocks. We compare the original equations of *MyD88* and *IRAK4* which have 1 and 2 incoming influences each.

$$x_{MyD88}(t+1) = x_{TLR5}(t)$$
$$x_{IRAK4}(t+1) = (\neg x_{MyD88}(t) \wedge x_{TICAM1}(t)) \vee (x_{MyD88}(t))$$

We see that the two variables are BBE because their update functions depend only on the BBE-equivalent variables *TLR5* and *MyD88*, respectively. For *IRAK4*, the three variables in the update function are BBE. Therefore, they have same value allowing us to simplify the update function to just *MyD88*. The ODEs obtained for the 2 variables are, where $x'_{\_}$ denotes the derivative of $x_{\_}$:

$$x'_{MyD88} = x_{TLR5} - x_{MyD88}$$
$$x'_{IRAK4} = x_{MyD88} + x_{TICAM1} - x_{MyD88} \cdot x_{TICAM1} - x_{IRAK4}$$

Given that all variables appearing in the equations are backward equivalent, the two equations coincide with the original ones when all variables have values either 0 or 1. However, they differ for non-integer values. For example, in case all variables have value 0.5, we get 0 for the former, and 0.25 for the latter.

5 Related Work

BN reduction techniques belong to three families according to their domain of reduction: (i) they reduce at syntactic level (i.e. the BN [34,48,39,3,32,41,51]), (ii) at semantic level (i.e. the STG [23,1]), or (iii) they transform BNs to other formalisms like Petri Nets [16,44] and ordinary differential equations [50] offering formalism-specific reductions. However, (semantic) STG-reduction does not solve the state space explosion whereas the transformation to other formalisms has several drawbacks as shown in Section 4.3.

Syntactic level reduction methods usually perform variable absorption [3,34,48,41] at the BN. BN variables can get absorbed by the update functions of their target variables by replacing all occurrences of the absorbed variables with their update functions. This method was first investigated in [34] wherein update functions are represented as ordinary multivalued decision diagrams. The authors consider multivalued networks with updates being applied asynchronously and iteratively implement absorption. The process, despite preserving steady states in all synchronization schemas [48], might lead to loss of cycle attractors in the synchronous schema. However, absorption of variables might lead to introduction of new attractors in the asynchronous case, i.e., by reducing the number of variables the number of attractors can stay the same or increase (attractors can split or new attractors can appear).

A similar study [48] presents a reduction procedure and proves that it preserves steady states. This procedure includes two steps. The first refers to the deletion of links between variables on their network structure. Deletion of pseudo-influences is feasible by simplifying the Boolean expressions in update functions. The second step of the procedure refers to the absorption of variables like in [34].

The difference between studies [48], [34] is that [48] exploits Boolean algebra instead of multivalued decision diagrams to explain absorption. Moreover, they refer only to Boolean networks, and do not consider any update schema. In studies [34,48,41], self-regulated BN variables (i.e. variables with a self-loop in the graphical representation) can not be selected for absorption. The inability to

absorb self-regulated variables is inherent in the implementation of absorption in contrast to our method where the restrictions are encoded by the user at the initial partition and self-regulated variables can be merged with other variables.

In [41] the authors presented a two step reduction algorithm. The first step includes the absorption of input variables with stable function and the second step the absorption of single mediator variables (variables with one incoming and outgoing edge in the signed interaction graph). The first step of the algorithm in [41] is equally useful and compatible with the first step of [48]. Moreover, if we combine the first steps of [48] and [41], we may achieve interesting reductions which exactly preserve all asymptotic dynamics.

The first steps of [48,41] affect only a BN property called *stability*. Stability is the ability of a BN to end up to the same attractor when starting from slightly different initial conditions. In [3], the authors introduced the decimation procedure -a reduction procedure for synchronous BNs- to discuss how it affects stability. The crucial difference between decimation procedure and BBE-reduction is that the first was invented to study stability whereas the latter was invented to degrade state space explosion. The decimation procedure is summarized by the following four steps: (i) remove from every update functions the inputs that it does not depend on, (ii) find the constant value for variables with no inputs, (iii) propagate the constant values to other update functions and remove this variable from the system, and (iv) if a variable has become constant, repeat from step (i). The study also refers to leaf variables because their presence does not play any role in the asymptotic dynamics of a BN. However, both leaf and fixed-valued variables affect stability. Overall, the decimation procedure exactly preserves the asymptotic dynamics of the original model since it throws out only variables considered as asymptotically irrelevant.

6 Conclusion

We introduced an automatic reduction technique for synchronous Boolean Networks which preserves dynamics of interest. The modeller gets a reduced BN based on requirements expressed as an initial partition of variables. The reduced BN can recover a pure part of the original state space and its trajectories established by the reduction isomorphism. Notably, we draw connections between the STG of the original and that of the reduced BN through a rigorous mathematical framework. The dynamics preserved are those wherein collapsed variables have equal values.

We used our reduction technique to speed-up attractor identification. Despite that the length of the preserved attractors is consistent in the reduced model, some of them may get lost. In the future, we plan to study classes of initial partitions that preserve all attractors. We have shown the analysis speed-ups obtained for attractor identification as implemented in the tool BoolSim [24]. In the future we plan to perform a similar analysis on a recent attractor identification approach from [21].

Our method was implemented in ERODE [10], a freely available tool for reducing biological systems. Related *quantitative* techniques offered by ERODE have been recently validated on a large database of biological models [37,38,5]. In the future we plan to extend this analysis considering also BBE. We also plan to investigate whether BBE can be extended in order to be able to compare different models as done for its quantitative counterparts [7,12].

Our method could be combined with most of the existing methods found in literature. Our prototype toolchain consists of several tools from the COLOMOTO interoperability initiative. We aim to incorporate our toolchain into the COLOMOTO Interactive Notebook [35], a unified environment to edit, execute, share, and reproduce analyses of qualitative models of biological networks.

Multivalued BNs, i.e. whose variables can take more than two activation values, are currently supported only via a *booleanization* technique [18,14] that might hamper the interpretability of the reduced model. In future work we plan to generalize BBE to support directly multivalued networks.

References

1. Bérenguier, D., Chaouiya, C., Monteiro, P.T., Naldi, A., Remy, E., Thieffry, D., Tichit, L.: Dynamical modeling and analysis of large cellular regulatory networks. Chaos: An Interdisciplinary Journal of Nonlinear Science **23**(2), 025114 (2013)
2. Biere, A., Biere, A., Heule, M., van Maaren, H., Walsh, T.: Handbook of Satisfiability: Volume 185 Frontiers in Artificial Intelligence and Applications. IOS Press, NLD (2009)
3. Bilke, S., Sjunnesson, F.: Stability of the Kauffman model. Physical Review E **65**(1), 016129 (2001)
4. Calzone, L., Tournier, L., Fourquet, S., Thieffry, D., Zhivotovsky, B., Barillot, E., Zinovyev, A.: Mathematical modelling of cell-fate decision in response to death receptor engagement. PLoS Comput Biol **6**(3), e1000702 (2010)
5. Cardelli, L., Pérez-Verona, I.C., Tribastone, M., Tschaikowski, M., Vandin, A., Waizmann, T.: Exact maximal reduction of stochastic reaction networks by species lumping. CoRR **abs/2101.03342** (2021), `https://arxiv.org/abs/2101.03342`
6. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Forward and backward bisimulations for chemical reaction networks. In: 26th International Conference on Concurrency Theory, CONCUR 2015, Madrid, Spain, September 1.4, 2015. pp. 226–239 (2015). https://doi.org/10.4230/LIPIcs.CONCUR.2015.226, `https://doi.org/10.4230/LIPIcs.CONCUR.2015.226`
7. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Comparing chemical reaction networks: A categorical and algorithmic perspective. In: Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016. pp. 485–494 (2016). https://doi.org/10.1145/2933575.2935318, `https://doi.org/10.1145/2933575.2935318`
8. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Efficient syntax-driven lumping of differential equations. In: Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software,

ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings. pp. 93–111 (2016). https://doi.org/10.1007/978-3-662-49674-9_6, https://doi.org/10.1007/978-3-662-49674-9_6

9. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Symbolic computation of differential equivalences. In: Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016. pp. 137–150 (2016). https://doi.org/10.1145/2837614.2837649, https://doi.org/10.1145/2837614.2837649
10. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Erode: a tool for the evaluation and reduction of ordinary differential equations. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 310–328. Springer (2017)
11. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Maximal aggregation of polynomial dynamical systems. Proceedings of the National Academy of Sciences **114**(38), 10029–10034 (2017)
12. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Comparing chemical reaction networks: A categorical and algorithmic perspective. Theor. Comput. Sci. **765**, 47–66 (2019). https://doi.org/10.1016/j.tcs.2017.12.018, https://doi.org/10.1016/j.tcs.2017.12.018
13. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Symbolic computation of differential equivalences. Theor. Comput. Sci. **777**, 132–154 (2019). https://doi.org/10.1016/j.tcs.2019.03.018, https://doi.org/10.1016/j.tcs.2019.03.018
14. Chaouiya, C., Bérenguier, D., Keating, S.M., Naldi, A., Van Iersel, M.P., Rodriguez, N., Dräger, A., Büchel, F., Cokelaer, T., Kowal, B., et al.: SBML qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools. BMC systems biology **7**(1), 1–15 (2013)
15. Chaouiya, C., Naldi, A., Thieffry, D.: Logical modelling of gene regulatory networks with ginsim. In: Bacterial Molecular Networks, pp. 463–479. Springer (2012)
16. Chaouiya, C., Remy, E., Thieffry, D.: Petri net modelling of biological regulatory networks. Journal of Discrete Algorithms **6**(2), 165–177 (2008)
17. De Moura, L., Bjørner, N.: Z3: An efficient smt solver. In: International conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 337–340. Springer (2008)
18. Delaplace, F., Ivanov, S.: Bisimilar booleanization of multivalued networks. BioSystems p. 104205 (2020)
19. Di Cara, A., Garg, A., De Micheli, G., Xenarios, I., Mendoza, L.: Dynamic simulation of regulatory networks using squad. BMC bioinformatics **8**(1), 462 (2007)
20. Drossel, B.: Random boolean networks. Reviews of nonlinear dynamics and complexity **1**, 69–110 (2008)
21. Dubrova, E., Teslenko, M.: A sat-based algorithm for finding attractors in synchronous boolean networks. IEEE/ACM transactions on computational biology and bioinformatics **8**(5), 1393–1399 (2011)
22. Fauré, A., Naldi, A., Lopez, F., Chaouiya, C., Ciliberto, A., Thieffry, D.: Modular logical modelling of the budding yeast cell cycle. Molecular bioSystems **5**, 1787–96 (2009 Dec 2009)
23. Figueiredo, D.: Relating bisimulations with attractors in boolean network models. In: International Conference on Algorithms for Computational Biology. pp. 17–25. Springer (2016)

24. Garg, A., Di Cara, A., Xenarios, I., Mendoza, L., De Micheli, G.: Synchronous versus asynchronous modeling of gene regulatory networks. Bioinformatics **24**(17), 1917–1925 (07 2008). https://doi.org/10.1093/bioinformatics/btn336, https://doi.org/10.1093/bioinformatics/btn336
25. Giacomantonio, C.E., Goodhill, G.J.: A boolean model of the gene regulatory network underlying mammalian cortical area development. PLOS Computational Biology **6**(9), 1–13 (09 2010). https://doi.org/10.1371/journal.pcbi.1000936, https://doi.org/10.1371/journal.pcbi.1000936
26. Grieco, L., Calzone, L., Bernard-Pierrot, I., Radvanyi, F., Kahn-Perles, B., Thieffry, D.: Integrative modelling of the influence of mapk network on cancer cell fate decision. PLoS Comput Biol **9**(10), e1003286 (2013)
27. Hopfensitz, M., Müssel, C., Maucher, M., Kestler, H.A.: Attractors in boolean networks: a tutorial. Computational Statistics **28**(1), 19–36 (2013)
28. Kauffman, S.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology **22**(3), 437 – 467 (1969)
29. Klamt, S., Haus, U.U., Theis, F.: Hypergraphs and cellular networks. PLoS computational biology **5**(5) (2009)
30. Klarner, H., Streck, A., Siebert, H.: Pyboolnet: a python package for the generation, analysis and visualization of boolean networks. Bioinformatics **33**(5), 770–772 (2017)
31. Naldi, A., Berenguier, D., Fauré, A., Lopez, F., Thieffry, D., Chaouiya, C.: Logical modelling of regulatory networks with ginsim 2.3. Biosystems **97**(2), 134–139 (2009)
32. Naldi, A., Monteiro, P.T., Chaouiya, C.: Efficient handling of large signalling-regulatory networks by focusing on their core control. In: International Conference on Computational Methods in Systems Biology. pp. 288–306. Springer (2012)
33. Naldi, A., Monteiro, P.T., Müssel, C., for Logical Models, C., Tools, Kestler, H.A., Thieffry, D., Xenarios, I., Saez-Rodriguez, J., Helikar, T., Chaouiya, C.: Cooperative development of logical modelling standards and tools with colomoto. Bioinformatics **31**(7), 1154–1159 (2015)
34. Naldi, A., Remy, E., Thieffry, D., Chaouiya, C.: Dynamically consistent reduction of logical regulatory graphs. Theoretical Computer Science **412**(21), 2207–2218 (2011)
35. Naldi, A., Hernandez, C., Levy, N., Stoll, G., Monteiro, P.T., Chaouiya, C., Helikar, T., Zinovyev, A., Calzone, L., Cohen-Boulakia, S., Thieffry, D., Paulevé, L.: The colomoto interactive notebook: Accessible and reproducible computational analyses for qualitative biological networks. Frontiers in Physiology **9**, 680 (2018). https://doi.org/10.3389/fphys.2018.00680, https://www.frontiersin.org/article/10.3389/fphys.2018.00680
36. Paige, R., Tarjan, R.E.: Three partition refinement algorithms. SIAM Journal on Computing **16**(6), 973–989 (1987)
37. Pérez-Verona, I.C., Tribastone, M., Vandin, A.: A large-scale assessment of exact model reduction in the biomodels repository. In: Computational Methods in Systems Biology - 17th International Conference, CMSB 2019, Trieste, Italy, September 18-20, 2019, Proceedings. pp. 248–265 (2019). https://doi.org/10.1007/978-3-030-31304-3_13, https://doi.org/10.1007/978-3-030-31304-3_13
38. Pérez-Verona, I.C., Tribastone, M., Vandin, A.: A large-scale assessment of exact model reduction in the biomodels repository. Theoretical Computer Science (2021)
39. Richardson, K.A.: Simplifying boolean networks. Advances in Complex Systems **8**(04), 365–381 (2005)

40. Rodríguez-Jorge, O., Kempis-Calanis, L.A., Abou-Jaoudé, W., Gutiérrez-Reyna, D.Y., Hernandez, C., Ramirez-Pliego, O., Thomas-Chollier, M., Spicuglia, S., Santana, M.A., Thieffry, D.: Cooperation between t cell receptor and toll-like receptor 5 signaling for cd4+ t cell activation. Science signaling **12**(577) (2019)
41. Saadatpour, A., Albert, R., Reluga, T.C.: A reduction method for boolean network models proven to conserve attractors. SIAM Journal on Applied Dynamical Systems **12**(4), 1997–2011 (2013)
42. Schwab, J.D., Kühlwein, S.D., Ikonomi, N., Kühl, M., Kestler, H.A.: Concepts in boolean network modeling: What do they all mean? Computational and Structural Biotechnology Journal **18**, 571 – 582 (2020). https://doi.org/https://doi.org/10.1016/j.csbj.2020.03.001, http://www.sciencedirect.com/science/article/pii/S200103701930460X
43. Shmulevich, I., Dougherty, E.R., Kim, S., Zhang, W.: Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks. Bioinformatics **18**(2), 261–274 (2002)
44. Steggles, L.J., Banks, R., Shaw, O., Wipat, A.: Qualitatively modelling and analysing genetic regulatory networks: a petri net approach. Bioinformatics **23**(3), 336–343 (2007)
45. Su, C., Pang, J.: Sequential control of boolean networks with temporary and permanent perturbations. arXiv preprint arXiv:2004.07184 (2020)
46. Thomas, R.: Regulatory networks seen as asynchronous automata: a logical description. Journal of theoretical biology **153**(1), 1–23 (1991)
47. Thomas, R.: Kinetic logic: a Boolean approach to the analysis of complex regulatory systems: proceedings of the EMBO course "formal analysis of genetic regulation", held in Brussels, September 6–16, 1977, vol. 29. Springer Science & Business Media (2013)
48. Veliz-Cuba, A.: Reduction of boolean network models. Journal of theoretical biology **289**, 167–172 (2011)
49. Wittmann, D.M., Krumsiek, J., Saez-Rodriguez, J., Lauffenburger, D.A., Klamt, S., Theis, F.J.: Transforming boolean models to continuous models: methodology and application to T-cell receptor signaling. BMC Systems Biology **3**(1), 98 (2009). https://doi.org/10.1186/1752-0509-3-98
50. Wittmann, D.M., Krumsiek, J., Saez-Rodriguez, J., Lauffenburger, D.A., Klamt, S., Theis, F.J.: Transforming boolean models to continuous models: methodology and application to t-cell receptor signaling. BMC systems biology **3**(1), 98 (2009)
51. Zañudo, J.G.T., Albert, R.: An effective network reduction approach to find the dynamical repertoire of discrete dynamic networks. Chaos: An Interdisciplinary Journal of Nonlinear Science **23**(2), 025111 (2013). https://doi.org/10.1063/1.4809777, https://doi.org/10.1063/1.4809777
52. Zhang, R., Shah, M.V., Yang, J., Nyland, S.B., Liu, X., Yun, J.K., Albert, R., Loughran, T.P.: Network model of survival signaling in large granular lymphocyte leukemia. Proceedings of the National Academy of Sciences **105**(42), 16308–16313 (2008)

A Proofs

Proof (Proof of Theorem 1). Let X_{R_1}, X_{R_2} be two BBE partitions that refine some other partition X_I that is not necessarily a BBE. We start by noting that $R = (R_1 \cup R_2)^* \subseteq I$ because $R_1, R_2 \subseteq I$, where asterisk denotes the transitive closure, while R_1, R_2 and I are equivalence relations underlying X_{R_1}, X_{R_2} and X_I, respectively. Hence, X_R is a refinement of X_I. We next show that X_R is a BBE partition. To this end, fix some $\mathbf{s} \in \mathbb{B}^n$ that is constant on R. Since $R_i \subseteq R$, this implies that $\mathbf{s}$ is constant on R_i which, in virtue of X_{R_i} being a BBE, implies that $F(\mathbf{s}) \in \mathbb{B}^n$ is constant on R_i. This implies that $F(\mathbf{s}) \in \mathbb{B}^n$ is constant on $R = (R_1 \cup R_2)^*$, thus showing that X_R is indeed a BBE partition. The overall claim follows by noting that the finiteness of X implies that there are finitely many BBE partitions X_{R_i} that refine any given partition X_I of X.

Proof (Proof of Theorem 2). Assume that G' denotes the coarsest BBE partition that refines some given partition G. Set $H_0 := G$ and define for all $k \geq 0$

$$H_{k+1} := \big(\{C_0 \mid C \in H_k\} \cup \{C_1 \mid C \in H_k\}\big) \setminus \{\emptyset\},$$

where C_0 and C_1 are as in Algorithm 1. Then, a proof by induction over $k \geq 1$ shows that (a) G' is a refinement of H_k and (b) H_k is a refinement of H_{k-1}, for all $k \geq 1$. Since G' is a refinement of any H_k, it holds that $G' = H_k$ if H_k is a BBE partition. Since X is finite, b) allows us to fix the smallest $k \geq 1$ such that $H_k = H_{k-1}$. This, in turn, implies that H_{k-1} is a BBE.

Proof (Proof of Lemma 1). The fact that $A \cap S_{|R} \neq \emptyset$ implies that there is at least one state $\mathbf{s} \in A$ that is constant on R, i.e., $\mathbf{s} \in A \cap S_{|R}$. For any such state $\mathbf{s}$, by the properties of BBE we have that any state $\mathbf{t}$ such that $\mathbf{s} \to^+ \mathbf{t}$ is also constant on R. Actually, it is trivial to show that $A = \{\mathbf{t} \mid \mathbf{s} \to^+ \mathbf{t}\}$. It immediately follows that $A \subseteq S_{|R}$.

Proof (Proof of Lemma 2). Follows readily from the definition of a BBE and m_R.

Proof (Proof of Theorem 3). The theorem trivially follows from Lemmas 1 and 2.

B Table of large-scale validation

We provide the table referenced in the Section 4.1 on large-scale validation of BBE. The table contains the results of BBE reduction on all the models from the GINsim repository. The first column contains the model identifier (MI). The second column contains the url to download the model and the third column displays the number of variables in the original BN. In the case of multivalued networks, the column contains the number of variables after booleanization. We denote with N_m, N_i the number of variables of the maximal and the ID reduced BN in the fifth and sixth column respectively. Note that when a BN has no input variables N_i and N_m coincide. The last two columns display the reduction ratios $r_i = N_i/N$, $r_m = N_m/N$ where N is the number of variables in the original BN.

MI	*GINsim repository URI*	N	N_i	N_m	r_i	r_m
B1	http://ginsim.org/node/225	128	107	95	0.836	0.742
B2	http://ginsim.org/node/225	110	103	91	0.936	0.827
B3	http://ginsim.org/node/87	60	57	52	0.95	0.867
B4	http://ginsim.org/node/173	53	46	39	0.868	0.736
B5	http://ginsim.org/node/225	42	37	29	0.881	0.690
B6	http://ginsim.org/node/78	40	31	29	0.775	0.725
B7	http://ginsim.org/node/227	33	27	25	0.818	0.758
B8	http://ginsim.org/node/191	32	32	31	1	0.969
B9	http://ginsim.org/node/227	28	25	20	0.893	0.714
B10	http://ginsim.org/node/97	26	23	4	0.885	0.154
B11	http://ginsim.org/node/144	24	23	9	0.958	0.375
B12	http://ginsim.org/node/126	24	21	4	0.875	0.167
B13	http://ginsim.org/node/102	23	22	8	0.957	0.348
B14	http://ginsim.org/node/39	20	15	13	0.75	0.65
B15	http://ginsim.org/node/160	18	18	8	1	0.444
B16	http://ginsim.org/node/35	18	17		0.944	
B17	http://ginsim.org/node/31	14	14	12	1	0.857
B18	http://ginsim.org/node/152	11	10	9	0.909	0.818
B19	http://ginsim.org/node/37	10	9	9	0.9	0.9
B20	http://ginsim.org/node/69	10	8	8	0.8	0.8
B21	http://ginsim.org/model/C_crescentus	9	7		0.778	
B22	http://ginsim.org/node/21	9	7		0.778	
B23	http://ginsim.org/node/214	6	2		0.333	
B24	http://ginsim.org/model/C_crescentus	5	1		0.2	
M1	http://ginsim.org/model/tcell-checkpoint-inhibitors-tcla4-pd1	218	201	136	0.922	0.623
M2	http://ginsim.org/node/229	133	126	122	0.947	0.917
M3	http://ginsim.org/node/178	107	93	59	0.869	0.551
M4	http://ginsim.org/node/185	103	97	52	0.941	0.505
M5	http://ginsim.org/model/SP	102	16		0.157	
M6	http://ginsim.org/node/194	83	79		0.951	
M7	http://ginsim.org/node/79	71	69	42	0.972	0.592
M8	http://ginsim.org/node/234	61	60	58	0.983	0.95
M9	http://ginsim.org/model/drosophila_mesoderm	57	57	11	1	0.192
M10	http://ginsim.org/node/69	56	56	50	1	0.893
M11	http://ginsim.org/model/EMT_Selvaggio_etal	56	56	43	1	0.768
M12	http://ginsim.org/node/229	53	51	50	0.962	0.943
M13	http://ginsim.org/node/21	50	49	41	0.98	0.82
M14	http://ginsim.org/node/180	48	35		0.729	
M15	http://ginsim.org/node/25	39	37	31	0.948	0.794
M16	http://ginsim.org/model/sex_determination_chicken	37	37	14	1	0.378
M17	http://ginsim.org/node/79	36	35	21	0.972	0.583
M18	http://ginsim.org/node/188	35	34	28	0.971	0.8
M19	http://ginsim.org/node/216	34	34	33	1	0.971
M20	http://ginsim.org/node/96	34	32	15	0.941	0.441
M21	http://ginsim.org/node/183	30	30	14	1	0.467
M22	http://ginsim.org/model/eggshell_patterning	24	23	12	0.958	0.5
M23	http://ginsim.org/node/41	21	21	18	1	0.857
M24	http://ginsim.org/model/sex_determination_mammals	19	19	12	1	0.632
M25	http://ginsim.org/node/109	19	19	7	1	0.368
M26	http://ginsim.org/model/SP	19	18	17	0.947	0.895
M27	http://ginsim.org/node/89	18	18	10	1	0.556
M28	http://ginsim.org/node/29	16	16	13	1	0.812
M29	http://ginsim.org/model/sex_determination_chicken	15	15	13	1	0.866
M30	http://ginsim.org/node/194	14	14	13	1	0.928
M31	http://ginsim.org/node/26	12	12	8	1	0.667
M32	http://ginsim.org/node/115	16	16	5	1	0.3125
M33	http://ginsim.org/node/220	10	10	8	1	0.8
M34	http://ginsim.org/model/eggshell_patterning	8	8	2	1	0.25
M35	http://ginsim.org/node/82	7	7	6	1	0.857
M36	http://ginsim.org/node/82	7	7	6	1	0.857
M37	http://ginsim.org/node/50	6	6	5	1	0.833

Table 2: Application of BBE to BNs from the GINsim model repository.

C Refined initial partitions for the selected case studies

In Section 4.2, we studied how BBE affects attractor analysis of three selected case studies. It is remarkable that attractor identification was infeasible for the largest TCR-TLR5 BN, whereas we identified its attractors in 30 seconds in its maximal reduction. However, the attractors identified may not be all the attractors of interest for the BN. Our crucial hypothesis is that one can specify alternative initial partitions that preserve more or discard some irrelevant attractors. We also expect that the reduction ratio of these alternative initial partitions lies between that of the ID and that of the maximal reduction (r_m, r_i).

Configuration In Sections C.1 , C.2 and C.3 , we provide a detailed description of the initial partitions that lead to the *refined reduced models*.

Results The results of the refined MAPK, the refined merged TCR-TLR, and the refined T-LGL reduced models are summarized in the following Table 3 . We present the number of variables (*Size*), the number of *Attractors*, and the time needed for reduction (*Reduction (s)*) and attractor identification (*Analysis (s)*).

Model	*Original model*			*Refined Reduced model*			
	Size	*Attractors*	*Analysis (s)*	*Reduction (s)*	*Size*	*Attractors*	*Analysis (s)*
MAPK Network	53	40	16.501	1.202	42	40	12.115
T-LGL	60	264	123.431	1,816	56	120	55.049
TCR-TLR merged	128	—*Time Out*—		2.096	98	8	9349.577

Table 3: The results of 3 case studies for the original and the refined reduced BNs

The refined reduced MAPK network consists of 42 variables but preserves all attractors in the original model. Notably, the reduced model has *80%* the size of the original. The refined reduced T-LGL results from the original after specifying two input variables in the same block of the initial partition. The merging of these two input variables is an immediate result of [52] and discards 144 attractors which are irrelevant for their analysis. Last but not least, the refined reduced TCR-TLR merged has 98 variables and 8 attractors-more than the maximal reduction of Table 1 . Note again that the attractor identification in the original BN is intractable.

Interpretation Overall, Table 3 illustrates the possibility of analyzing large BNs by defining alternative initial partitions than the two considered in Section 4.1. Alternative reductions may provide fruitful insights and identify crucial properties of the underlying system. The size of the refined reduced model lies between the size of the input-distinguished and the maximal reduced model in all three models.

The initialization of Algorithm 1 provides also a framework to specify desires and limitations. Desires refer to the preserved properties with respect to the original model whereas limitations refer to variable perturbation. If such a variable get merged then its perturbation will indicate subsequent perturbation to all the variables that belong to its class. Consequently, variables that are amenable to perturbation, should be kept in singleton blocks of the initial partition. To this end, we can construct empirical initial partitions that (i) preserve attractors, (ii) discard some of them, or (iii) isolate in singleton blocks variables which are amenable to perturbation.

C.1 T Cell and Toll-like Receptor (TCR-TLR) merged signalling BN

In this Section, we exploit the results from the maximal and the ID reduction to obtain two refined reduced BNs of the TCR-TLR merged BN. This BN refers to the T cell receptors and their responsibility for the activation of T cell ([40], Fig. 9). The authors generated logical models for the TCR and the TLR5 signalling pathways, and merged them by considering their cross interactions. The original model contains 128 variables, fact that renders its analysis intractable. In order to experimentally validate the correctness of their new merged BN, they considered asynchronous update schema and performed reduction with absorption [34]. Absorption does not guarantee preservation of all asymptotic dynamics. It has been proven [48,34,41] that preserves only steady states and may cause spurious cyclic attractors when applied to asynchronous dynamics. When applied to synchronous dynamics, this method may also degenerate cyclic attractors. BBE-reduction maintain the lengths of the preserved attractors according to Theorem 3.

ID reduction The application of ID reduction to the merged model resulted in 10 equivalence classes displayed in Fig. 8 . We also display them in Fig. 9 with different colors for each class: Backward equivalent variables are represented with colored boxes, and colored boxes that belong to the same equivalence class have the same background. Variables in white background belong to singleton classes. The ID reduced BN is still huge (116 variables) and the attractor identification is intractable.

{*IRAK4*, *PIK3AP1*}	{*GRAP2*, *MAP4K1*}
{*TICAM1*, *MyD88*}	{*MKNK1*, *RPS6KA5*}
{*Foxo1*, *BAD*, *GSK3B*, *CDKN1A*}	{*MAPKAPK2*, *mTOR*}
{*MAP2K3*, *MAP2K7*}	{*Camk2*, *Camk4*}
{*Cyc1*, *CTNNB1*}	{*DAG*, *IP3*}

Fig. 8: The equivalence classes of the input-distinguished reduction

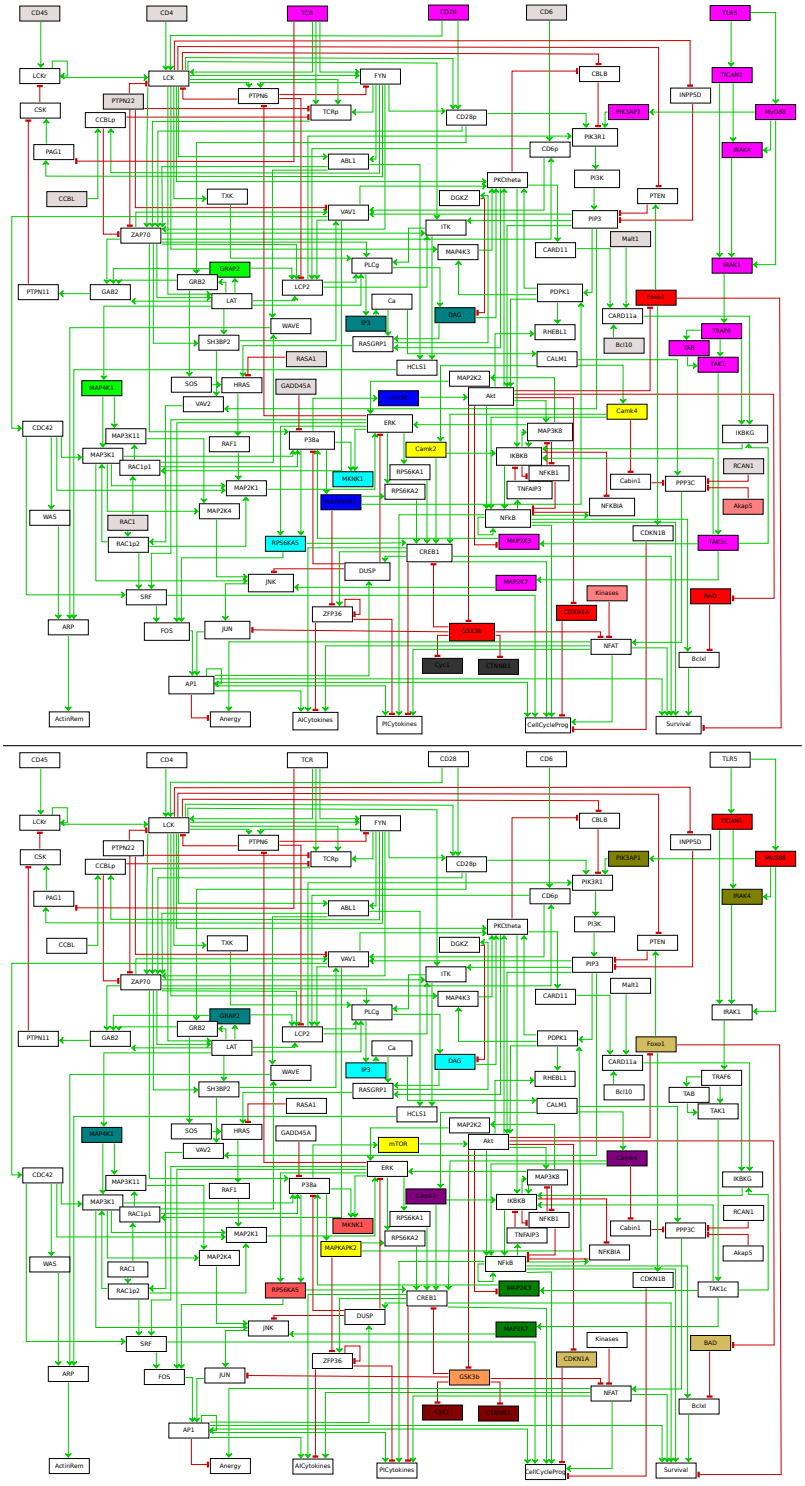


Fig. 9: Up: The TCR-TLR merged BN with input-distinguishing BBE-variables having the same background color.Bottom: The TCR-TLR merged BN with maximal-reduction BBE-variables having the same background color. Variables with white backround belong to singleton classes.

Maximal reduction However, the maximal reduced BN contains *95* variables and the attractor identification is feasible in *29.336* seconds. Fig. 9 displays the following BBE-equivalence classes with different colors:

*{*CD45*, *CD4*, *CD6*, *CCBL*, *PTPN22*, *Malt1*, *Bcl10*, *RCAN1*, *RAC1*, *RASA1*, *GDD45A*}
{*MAPKAPK2*, *mTOR*}
{*MKNK1*, *RPS6KA5*}
{*DAG*, *IP3*}
{*Camk2*, *Camk4*}

***{*TCR*, *CD28*, *TLR5*, *TICAM1*, *PIK3AP1*, *MyD88*, *IRAK1 TRAF6*, *TAB*, *TAK1*, *TAK1c*, *IRAK4*, *MAP2K3*, *MAP2K7*}
** {*Kinases*, *Akap5*}
{*Foxo1*, *BAD*, *GSK3B*, *CDKN1A*}
{*GRAP2*, *MAP4K1*}
{*Cyc1*, *CTNNB1*}

Fig. 10: The equivalence classes of the maximal reduction

The maximal reduction splits input variables into 3 classes: The first class (* in Fig. 10) contains all variables with stable update function which equals to *true*. The second class (**) contains all variables with stable update functions that equals to *false*. The third class (***) contains all variables with identity update function (*TCR*, *CD28*, *TLR5*) and all variables BBE-equivalent variables with them.

Refined Reduction We now consider two alternative initial partitions, initialize Algorithm 1 with them, and gain deeper insights in the underlying model. The first initial partition is constructed as follows:

- two of the inputs with identity function, *TCR* and *CD28*, are kept in singleton blocks,
- variables with stable function *true* belong to one block,
- variables with stable function *false* belong to another block, and
- we define one more block containing *TLR5* and all variables that belong to its equivalent class in the case of maximal reduction i.e. {*TLR5*, *TICAM1*, *PIK3AP1*, *MyD88*, *IRAK1*, *TRAF6*, *TAB*, *TAK1*, *TAK1c*, *IRAK1*, *MAP2K3*, *MAP2K7*}-the blue chain of variables (Fig. 9 top).

We call the reduced BN obtained by the first initial partition *refined reduced model I*.

The second initial partition that we consider is similar to the first but one subtle differentiation: the variable *MAP2K3* is kept in singleton block. The reduced BN that results from this initial partition, is called *refined reduced model II*. The results of our study in this model is summarized in Table 4 . We present the number of variables (*Size*), the number of *Attractors*, and the time needed for reduction (*Reduction (s)*) and attractor identification (*Analysis (s)*).

Interpretation As we have seen before, attractor identification is intractable in the original and the ID-reduced BN whereas we can identify two attractors in the case of maximal reduction. Attractor identification in the refined reduced

Model	*Size*	*Attractors*	*Analysis (s)*	*Reduction (s)*
Original	128	—*Time Out*—		-
Input-distinguished	116	—*Time Out*—		2.058
Refined Reduced II	98	8	9349.577	2.096
Refined Reduced I	97	8	1103.912	1.833
Maximal	95	2	29.336	1.958

Table 4: The results of the TCR-TLR merged BN for different reduced versions of the original model.

models is still feasible wherein we find more attractors than in the case of maximal reduction. We should highlight that reducing the TCR-TLR merged BN by just one variable decreases attractor identification time by several orders of magnitude (see *Analysis time* in the case of *Refined Reduced Models*). The computation of the BBE-reduced BN took less than *3* seconds in all cases. To sum up, initializing Algorithm 1 with alternative initial partitions derived from the results of the maximal and ID reduction enables us to explore richer behaviours of the original model.

C.2 Mitogen-Activated Protein Kinase (MAPK) network

In this Section, we obtain a refined reduced model of the MAPK BN using results from the maximal and the ID reduced model. The original MAPK BN [26] consists of *53* variables, *4* inputs and has *40* attractors. We performed ID BBE-reduction and found the following equivalence classes: $\{JNK, p38\}$, $\{SMAD, TAK1\}$, $\{ATF2, JUN, MAX, PPP2CA\}$, $\{ELK1, MSK\}$, $\{RSK, SPRY\}$. The classes are displayed in the up part of Fig. 11 : Backward equivalent variables are represented with colored boxes and colored boxes that belong to the same class have the same background. Variables in white background belong to singleton classes.

ID Reduction The ID reduced MAPK BN has *46* variables and *40* attractors. Note that all attractors are preserved. This is a trivial consequence from the fact that (i) the number of attractors is the same, and (ii) the STG of the reduced BN is a subgraph of the STG of the original BN (isomorphism Lemma 2). The BBE-reduction is consistent with [11], where the authors transformed the BN to a system of ordinary differential equations, and reduced with backward differential equivalence.

Maximal Reduction The bottom part of Fig. 11 displays the MAPK BN and its equivalence classes after performing the maximal reduction. Algorithm 1 found the following equivalence classes: $\{JNK, p38\}$, $\{SMAD, TGFBR, ATM, TAOK, EGFR\_stimulus, FGFR3\_stimulus, TGFBR\_stimulus, DNA\_damage, TAK1\}$, $\{ATF2, JUN, MAX, PPP2CA\}$, $\{ELK1, MSK\}$, $\{RSK, SPRY\}$. BoolSim computes *17* attractors in this case which means that the number of attractors is not preserved. In contrast with [34], the preserved attractors are pure in the original

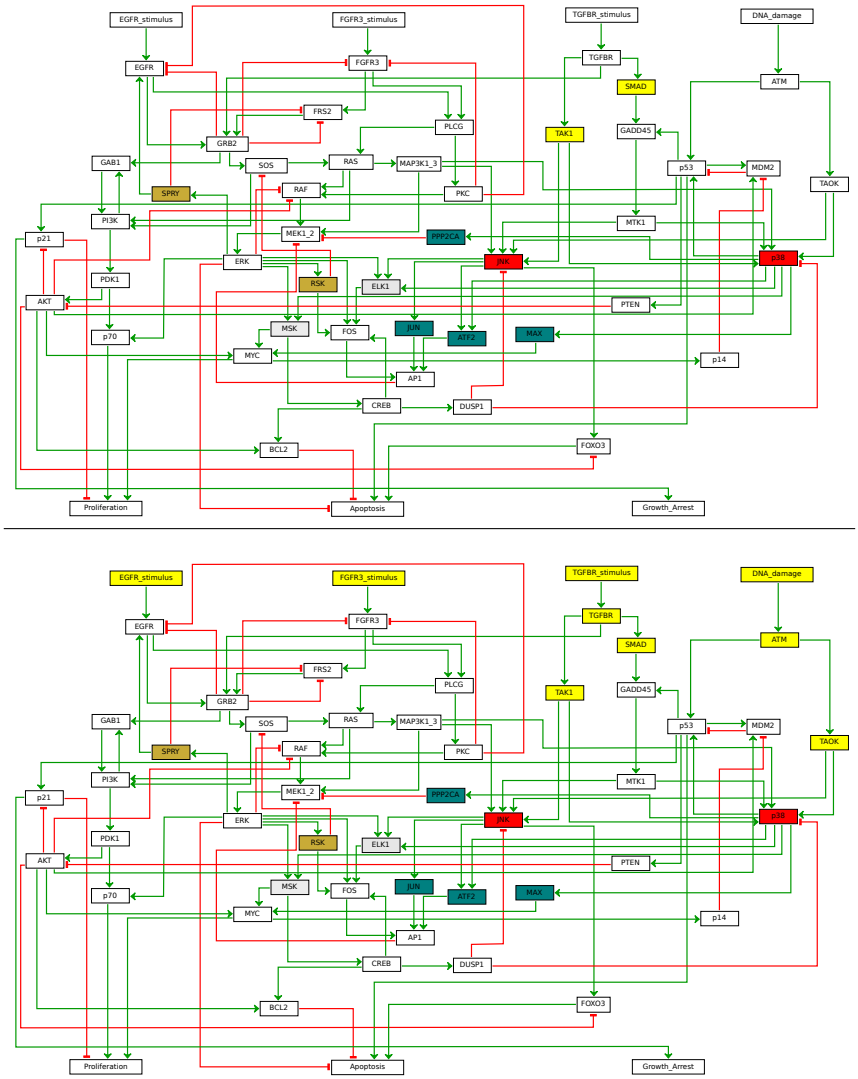


Fig. 11: Up: The MAPK BN with input-distinguishing BBE-variables having the same background color. Bottom: The MAPK BN with maximal-reduction BBE-variables having the same background color. Variables with white backround belong to singleton classes.

network in the sense that the isomorphism of Lemma 2 translates the attractors of the reduced to the original BN.

Refined Reduction Based on observations gained from the maximal and the ID reduction, we specify the following partition: {*EGFR_stimulus*}, {*FGFR3_stimulus*}, {*TGFBR_stimulus*, *TGFBR*, *TAK1*, *SMAD*}, {*DNA_damage*, *ATM*, *TAOK*}, and one block containing all the remaining variables. In other words, we keep 2 of the inputs ({*EGFR_stimulus*}, {*FGFR3_stimulus*}) still in singleton sets, while we define two more blocks with each one containing one input and the input's BBE-variables found in the maximal reduction. We expect that the reduced BN, which now contains *42* variables (*79*, *25*% of the original size), preserves more properties. Indeed, the refined reduced BN has all *40* attractors of the original BN. The results of our study in this model is summarized in the following Table 5 . We present the number of variables (*Size*), the number of *Attractors*, and the time needed for reduction (*Reduction (s)*) and attractor identification (*Analysis (s)*).

Model	*Size*	*Attractors*	*Analysis (s)*	*Reduction (s)*
Original	53	40	16.501	-
Input-distinguished	46	40	14.480	0.848
Refined Reduced	42	40	12.115	1.202
Maximal	39	17	2.471	1.018

Table 5: The results of the MAPK BN for different reduction versions of the original model.

C.3 T cell granular lymphocyte (T-LGL) leukemia BN

T-LGL BN was originally introduced in [52] and refers to the disease T-LGL leukemia which features a clonal expansion of antigen-primed, competent, cytotoxic T lymphocytes (CTL). The T-LGL BN is a signalling pathway, constructed empirically through extensive literature review, and determines the survival of CTL. The ID reduction and the maximal reduction are depicted in the top part and the bottom part of Fig. 12 respectively. The original BN consists of 60 variables, and has 264 attractors.

In the case of ID reduction, the variables *FasT*, *A20*, *TNF* and *RANTES* are BBE-equivalent so we can collapse them into a single variable. The ID reduced BN has 57 variables, and 264 attractors. Since the number of attractors is the same, and the STG of the reduced BN is a subgraph of the STG of the original BN, the asymptotic dynamics are preserved. The bottom part of Fig. 12 refers to the maximal BBE. In this case, we have two equivalence classes: the one found in ID BBE, and one consisted of all the input variables. On the other hand, the maximal reduced BN has 52 variables and 6 attractors. This means that some attractors are lost.

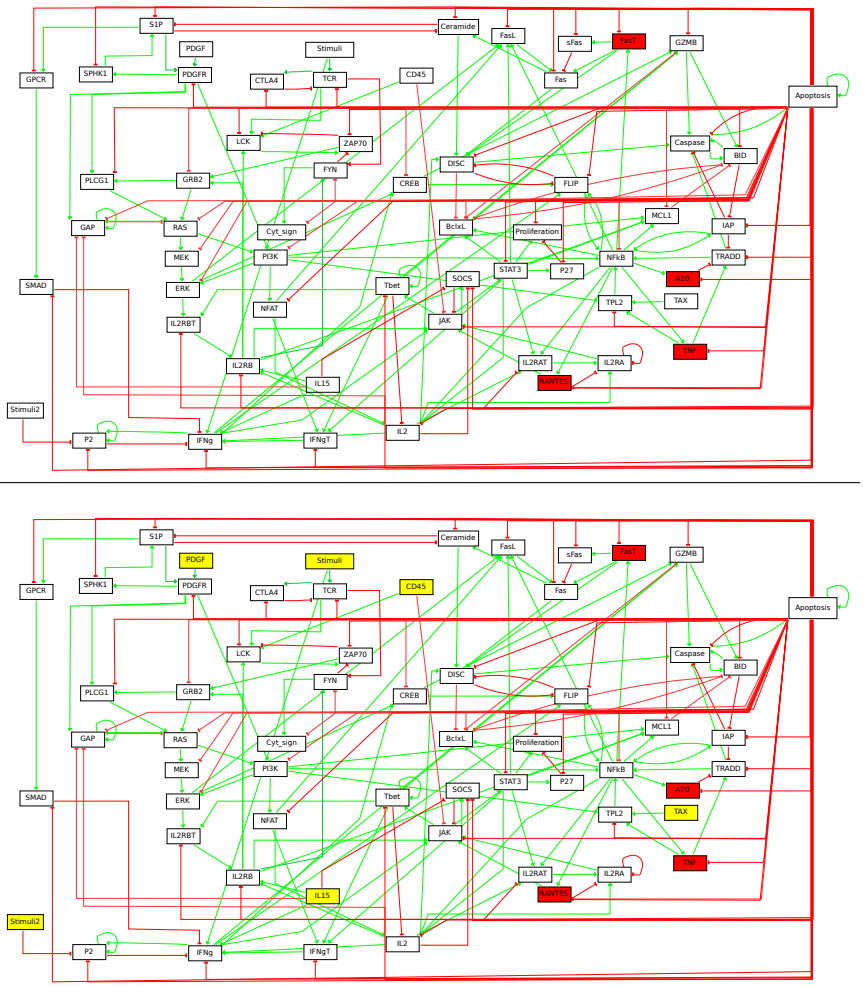


Fig. 12: Up: The T-LGL BN with input-distinguishing BBE-variables having the same background color. Bottom: The T-LGL BN with maximal-reduction BBE-variables having the same background color. Variables with white backround belong to singleton classes.

In [52], the authors considered the asynchronous schema and presented a variable specified analysis. Specifically, their analysis determined which variables are sufficient to induce all of known signalling abnormalities in leukemic T-LGL, which variables are important for the survival of leukemic T-LGL, and which variables are constantly active in leukemic T-LGL. Notably, permanent activation of the variables $IL-15$ and $PDGF$ is sufficient to produce all of the known deregulations and signalling abnormalities. For this reason, we consider as reasonable initial partition one wherein $IL-15$ and $PDGF$ belong to the same block, other input variables belong to singleton blocks, and non-input variables belong to one block. In this case, the refined reduced BN has 56 variables and 120 attractors. In contrast with the maximal reduced and the original BN which have 264 attractors, the BN reduced with this reasonable initial partition discards 144 attractors which are irrelevant for this kind of analysis. The results of our study on this model is summarized in the following Table 6 . We present the number of variables (*Size*), the number of *Attractors*, and the time needed for reduction (*Reduction (s)*) and attractor identification (*Analysis (s)*).

Model	*Size*	*Attractors*	*Analysis (s)*	*Reduction (s)*
Original	60	264	123.431	-
Input-distinguished	57	264	85.999	0.843
Refined Reduced	56	120	55.049	1.816
Maximal	52	6	2.489	0.999

Table 6: The results of the T-LGL BN for different reduction versions of the original model.

D Speed-ups on STG generation in BBE-reduced models

Hypothesis We hope to drastically reduce the time needed for STG generation. Furthermore, we claim that our technique may be utilized for STG visualization since reducing a BN only by one variable results in reducing its corresponding STG by *50*%.

Configuration We reduced the original BN with both ID and maximal BBE-reduction. We observed that PyBoolNet failed to generate the STG of BNs that have more than 25 variables. Hence, we restricted our experiments to all BNs with less variables. PyBoolNet generates the STG within several minutes for BNs between 21 and 25 variables, and within a minute for BNs with up to 20 variables. We did not consider BNs with less than 9 variables since the generation and visualization of the full STG is feasible and computationally costless. We present the results in Table 7:

Model	*Original model*		*Input-distinguished Reduced model*			*Maximal Reduced model*		
	Size	*STG generation(s)*	*Reduction (s)*	*Size*	*STG generation(s)*	*Reduction (s)*	*Size*	*STG generation(s)*
B7	33	*out of memory*	0.585	27	*out of memory*	0.608	25	*out of memory*
B9	28	*out of memory*	0.449	25	*out of memory*	0,416	20	52.8
B10	26	*out of memory*	0.227	23	457	0.145	4	0.006
B11	24	984	0.243	23	475	0.207	9	0,280
B12	24	987	0.349	21	102	0.121	4	0.050
B13	23	455	0.302	22	226	0.176	8	0.164
B14	20	55.6	0.497	15	2.11	0.408	13	0.302
B15	18	11.6	0.209	18	11.6	0,182	8	0.007
B16	18	14.300	*—NO INPUTS—*			0.449	17	6.760
B17	14	0,867	0.267	14	0.867	0.389	12	0.169
B18	11	0.072	0.327	10	0.064	0.214	9	0.065
B19	10	0.044	0.228	9	0.016	0.303	9	0.016
B20	10	0.172	0.283	8	0.044	0.202	8	0.044
B21	9	0.015	*—NO INPUTS—*			0.279	7	0.005
B22	9	0.025	*—NO INPUTS—*			0.237	7	0.003
M9	57	*out of memory*	0.791	57	*out of memory*	0.260	11	0.233
M16	37	*out of memory*	0.907	37	*out of memory*	0.454	14	1.360
M17	36	*out of memory*	0.413	35	*out of memory*	0.516	21	136
M20	34	*out of memory*	0.364	32	*out of memory*	0.383	15	2.68
M21	30	*out of memory*	0.421	30	*out of memory*	0.238	14	1.37
M22	24	1212	0.251	23	1043	0.219	12	0.172
M23	21	130	0.273	21	130	0.326	18	14.6
M24	19	31	0.109	19	31	0.153	7	0.463
M25	19	28.3	0.210	19	28.3	0.243	12	0.609
M26	19	30.1	0.249	18	13.9	0.356	17	6.320
M27	18	14.2	0.161	18	14.2	0.194	10	0.260
M28	16	3.34	0.189	16	3.34	0.266	13	1.54
M29	16	3.15	0.101	16	3.15	0.096	5	0.028
M30	15	1.59	0.187	15	1.59	0.235	13	0.303
M31	14	0.883	0.178	14	0.883	0.203	13	0.444
M32	12	0.156	0.168	12	0.156	0.137	8	0.010
M33	10	0.032	0.098	10	0.032	0.044	8	0.007

Table 7: Time needed for model reduction and STG generation of the original and the reduced BN. The running times are coming from one run and the computation of the BBE-reduced BNs take no more than 1 second in the worst cases.

Results PyBoolNet failed to generate the STG of the original B9 [4]. This was done within a minute after applying maximal BBE-reduction. For BNs between 20 and 25 variables, our method drastically decreased the STG generation time: The STG of the ID BN needs on average *25*% of the time for the generation of the full STG, and the maximal BN needs less than *1*% of the time needed for the generation of the full STG. For BNs with less than 20 variables the reduction may be computationally effective in several cases (see B15 and B16 in Table 7).

Interpretation We should note that our method (i) may render the analysis of large BN models tractable in many cases (like B9, M9, and M21) and (ii) facilitates STG visualization. BBE-reduction constitutes a useful method for generating pure segments of the original state space. According to the isomorphism Lemma 2, the STG of the reduced BN constitutes a subgraph of the STG of the original BN resulting from it after the collapse of a BBE-class into one variable component. In other words, our reduction method provides a pure image of the

state space of the original BN. We utilize these results in Section 4.2 wherein we conduct an attractor based analysis.

B PAPER II: Reducing Boolean Networks with Backward Boolean Equivalence

Submitted to journal BMC Bioinformatics

RESEARCH

Reducing Boolean Networks with Backward Equivalence

Georgios A. Argyris[1], Alberto Lluch Lafuente[1], Mirco Tribastone[2], Max Tschaikowski[3] and Andrea Vandin[4,1*]

Abstract

Background: Boolean Networks (BNs) are a popular dynamical model in biology where the state of each component is represented by a variable taking binary values that express, for instance, activation/deactivation or high/low concentrations. Unfortunately, these models suffer from the state space explosion, i.e., there are exponentially many states in the number of BN variables, which hampers their analysis.

Results: We present Boolean Backward Equivalence (BBE), a novel reduction technique for BNs which collapses system variables that, if initialized with same value, maintain matching values in all states. A large-scale validation on 86 models from two online model repositories reveals that BBE is effective, since it is able to reduce more than 90% of the models. Furthermore, on such models we also show that BBE brings notable analysis speed-ups, both in terms of state space generation and steady-state analysis. In several cases, BBE allowed the analysis of models that were originally intractable due to the complexity. On two selected case studies, we show how one can tune the reduction power of BBE using model-specific information to preserve all dynamics of interest, and selectively exclude behavior that does not have biological relevance.

Conclusions: BBE complements existing reduction methods, preserving properties that other reduction methods fail to reproduce, and vice versa. BBE drops all and only the dynamics, including attractors, originating from states where BBE-equivalent variables have been initialized with different activation values The remaining part of the dynamics is preserved exactly, including the length of the preserved attractors, and their reachability from given initial conditions, without adding any spurious behaviours. Given that BBE is a model-to-model reduction technique, it can be combined with further reduction methods for BNs.

Keywords: Boolean Network, Model reduction, State-space generation, Attractors analysis, Partition refinement

*Correspondence:
andrea.vandin@santannapisa.it
[4]Department of Excellence EMbeDS and Institute of Economics, Sant'Anna School for Advanced Studies, Pisa, Italy
[1]Department of Applied Mathematics and Computer Science, Technical University of Denmark, Lyngby, Denmark
Full list of author information is available at the end of the article

Background

Boolean networks (BNs) are a popular model in systems biology where the dynamics is qualitatively associated with two levels. These may express, for instance, on/off behavior in gene regulation or high/low concentrations of molecular compounds [1]. In a BN, the state is defined as a vector of Boolean variables, each representing a distinct component of the system under consideration. The time evolution of each variable is governed by an update function, i.e., a Boolean expression that encodes how the other variables affect the change of state at each (discrete) time step [2, 3, 4, 5]. In the main text we focus on synchronous BNs whereby the next state is obtained by applying all the update functions to the activation values of the current state. However, in the supplementary material we show how our approach can be applied also to BNs with partially asynchronous update schema (see, e.g., the *priority classes* supported by the popular tool GINsim [6] as described in [7]).

From a computational viewpoint, BNs are challenging to analyze. For example, the state space of the network, known as the state transition graph (STG), has exponential size in the number of variables. Thus, a full enumeration of the state space is possible only for networks of limited size. Another relevant type of analysis concerns the computation of attractors, i.e., those sets of states toward which the system tends to evolve and remain [8, 9]; these are often associated with biologically intelligible conditions of the system under study such as cell differentiation [3, 10]. Attractor identification is NP-hard [11] and, even if efficient tools have been developed [12], they do not scale well for large BNs.

These computational difficulties have motivated the development of reduction methods to ease BN analysis. Available techniques can be classified in three families according to the type of reduction: (i) by reasoning directly on the BN structure [4, 13, 14, 15, 16]; (ii) by reducing the underlying STG [5, 17]; (iii) by transforming a BN into other formalisms for which specific reduction techniques are available [18, 19]. The latter two classes suffer two main limitations. First, STG-based reductions are still subject to state space explosion since they require the full enumeration of the state space to start with. Second, reductions via other formalisms may not be complete in the sense that the dynamics of the original BN and of the transformed model are not equivalent, hence some reductions may be missed (see Additional file 2).

In the case of reduction methods at the BN level, popular examples are based on the notion of *variable absorption*, proposed originally in [14, 15]. The main idea is that certain BN variables can get removed by replacing their occurrences with their update functions. This is based on the assumption that those variables evolve over time scales that justify that they can be updated first in the model. Other methods remove *output/leaf* variables [4, 13] (variables that do not appear in the update functions of other variables) or *frozen* ones (variables that stabilize to the same value after some iterations independently of the initial conditions) [16].

Here we present a complementary type of reduction method based on the computation of a partition of the variables in the BN, whereby the future dynamics of variables in a block of the partition are equal whenever they start from the same condition. This can be convenient, for instance, if one is interested in studying the dynamics due to simultaneous activation or deactivation of groups of variables [20] (see also the case studies presented in the *Results and discussion* section). We call this kind of relation a *Boolean backward equivalence* (BBE) because it is defined analogously to the notion of backward bisimulation for Markov chains [21], more recently extended to chemical reaction networks [19, 22] and ordinary differential equations [23]. Recently, it has been shown how this backward notion can be given also for linear differential algebraic equations (DAE) [24, 25]. Using DAE terminology, such backward notion has been related to the preservation of invariant subspaces.

Every reduction technique comes with its own intuitive interpretation. For example, if we consider variable absorption mentioned above, it is intuitively based on the idea of fast/slow decomposition: it is biologically plausible to absorb a variable in the update function of another when the former fires faster than the latter. BBE is based on the following three orthogonal considerations:

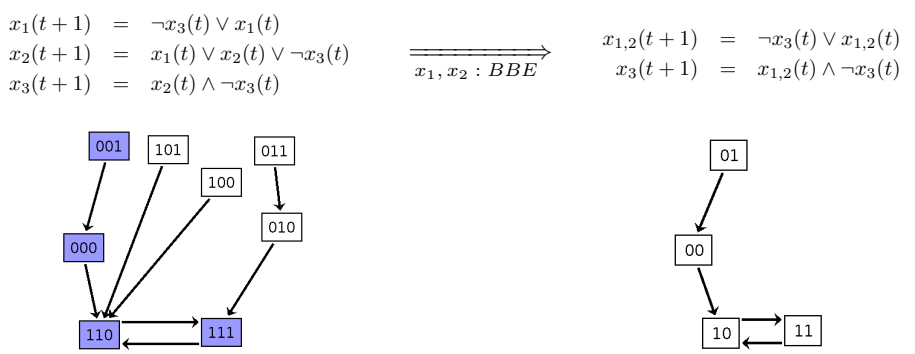


Figure 1: Boolean backward equivalence shown on a simple example. (Top-left) BN with three variables denoted by x_1, x_2, and x_3. (Bottom-left) The underlying STG. Each node is labelled by a vector that defines the state of each variable; a directed edge denotes a transition from a source state to a target state by a synchronous application of the update functions. States 110 and 111 form an attractor. (Top-right) Variables x_1 and x_2 can be shown to be BBE-equivalent by inspecting their update functions. If they have the same value in a state, i.e. $x_1(t) = x_2(t)$, then they will be equivalent for all successor states since $x_2(t+1) = x_1(t) \vee x_2(t) \vee \neg x_3(t) = x_1(t) \vee x_1(t) \vee \neg x_3(t) = x_1(t) \vee \neg x_3(t) = x_1(t+1)$. Based on this, a reduced BN can be obtained by considering a representative variable for each block and rewriting the corresponding update functions in terms of those representatives (here the representative variable is denoted by $x_{1,2}$). (Bottom-right) The underlying STG agrees with the original one on all states that have equal values for variables in the same block (purple nodes in bottom-left panel). Instead, any other state (i.e. those where variables in the same BBE block have different value), is removed. The criteria for BBE only involve checks for the update functions of the original model, such that the generation of original STG can be circumvented.

- BBE allows the modeler to discover chains of variables that, under some initialization conditions, describe the same dynamics. This might be interesting, e.g., to estimate the quality of a model: large BBE reductions might signal excessive redundancy in the model.
- As mentioned above, the modeler might be interested only in dynamics where two or more variables have simultaneous (de)activation value (see, e.g., [20]). The T-LGL case study in section *Results and discussion* further discusses this.
- In [23], it has been shown that this backward notion corresponds to Cardelli's *emulation* [26] which enables to relate a complex model with a simpler one. Interestingly, [26] discusses how emulation can be given an evolutionary interpretation. In fact, an original model can express all the dynamics of the reduced model. In addition, the original model can also express all additional dynamics coming from states where variables related by emulation have different activation values (not permitted in the reduced model because variables related by emulation get collapsed in the same reduced one). Given this richer dynamics of the original model, Cardelli uses selected case studies in [26] to

argue how the original model can be seen as an evolved version of the reduced one. We do not further investigate this aspect for BBE. However, given that BBE is based as well on the mentioned backward notions, it is not surprising that there exists a similar relation among the dynamics expressed by the original and reduced model (cf Fig. 1).

The criteria for a candidate partition of variables to be a BBE are encoded into a *satisfiability* problem over the expressions of the BN's update functions: we synthesise a Boolean expression involving BN variables and check whether there exists at least one combination of truth values for the variables that makes such expression true. This type of test can be effectively implemented using tools known as SAT solvers [27].

If a partition is a BBE, a reduced BN can be obtained by choosing and maintaining only a representative variable for each partition block, and renaming all variables in the remaining update functions with the representative one from their block. The STG of the reduced network exactly preserves the original dynamics for all states that have equal values across variables in the same block (Fig. 1). Importantly, however, the reduction method does not require the generation of the original STG, making it possible to obtain a reduced STG also from instances that would not be analyzable due to their massive size.

A crucial property satisfied by BBE is that there exists a *maximal* reduction for each BN, i.e., the coarsest BBE partition. This can be computed using a partition-refinement algorithm in a similar fashion as in Markov chains [28], reaction networks [22] and differential equations [23]. The algorithm essentially builds upon a fundamental result in computer science to prove equivalences in formal languages [29]. Given an initial partition of variables, the algorithm splits the blocks of the partition to compute its coarsest refinement that satisfies the BBE criteria. Thus, the maximal reduction is obtained when all variables are in the same unique block of the initial partition. However, the possibility of arbitrarily choosing the initial partition unlocks model-specific reduction queries that preserve the dynamics of user-defined variables. For example, in typical BN models of signaling pathways [30, 31], certain variables may represent the input signals to upstream components such as receptors. Formally, inputs may be detected because their update functions are constants that represent the values of such inputs. In this case, a possibly more biologically relevant initial partition may separate inputs from the other variables, obtaining *input-separated* (IS) reductions.

Our partition-refinement algorithm takes a polynomial number of steps as a function of the number of BN variables. At each iteration, it queries a SAT solver to check for the BBE criteria. If the query is satisfiable, i.e., the current partition is not a BBE, the returned assignment is used to split the current partition and perform another iteration; if the query is unsatisfiable, it returns the current partition as the coarsest BBE refinement of the initial one. Interestingly, although the algorithm is theoretically as complex as SAT solving, it behaves effectively in practice. Using a prototype implementation available within the software tool ERODE [32], we demonstrate its performance on a large-scale validation across 86 BN models from two well-known repositories [33, 6]. We show that almost all BNs can be reduced by BBE, providing speed-ups for the computation of STGs and attractors by more

than three orders of magnitude. In some cases, BBE could render the analysis feasible in instances that originally issued out-of-memory errors or that were stopped after long time outs. This comes at the cost that part of the original dynamics is lost. In particular, in the STG we preserve all and only the states where variables within the same BBE-block have the same value, and transitions among them. From this, and from the properties of BBE, we also get that the method preserves all and only the attractors containing at least one preserved state. This confirms that BBE is complementary to existing reduction techniques for BNs. Indeed, in several areas of science and engineering, it is common to have reduction techniques that:

- Preserve all dynamics but might add spurious ones. An example is [14] which preserves all attractors but might create new spurious ones. These often come with the name of *over-approximations* (this is because, e.g., [14] might over-approximate the set of attractors of a model by computing a larger set containing all original ones, plus some spurious ones;
- Do not preserve all the dynamics, but guarantee to not add spurious ones, like BBE. These often come with the name of *under-approximations* (this is because, e.g., BBE might under-approximate the set of attractors of a model by computing a smaller set containing only original ones, but potentially not all).

These two families of techniques are not comparable. They might be jointly used to obtain upper-bounds (the case of [14]), and lower-bounds (the case of BBE) on the actual number of attractors in a model.

This paper extends the previous conference version [34]. All numerical experiments have been redesigned by adding an additional model repository, by performing a large-scale validation of the analysis speed-ups offered by BBE, and by considering a more recent and efficient tool for identification of attractors. We have also performed a new large-scale validation on randomly generated BNs. Finally, we have generalised the theory to support also BNs with partially asynchronous update schema, and we have added a new case study considering one such BN.

Methods

Here we explain the key steps of the reduction procedure on the BN in Fig. 2. Its customary graphical representation allows one to distinguish different kinds of variables depending on whether they appear in the update functions of other variables (as indicated by the green arrows). In the example, *TLR5* can be interpreted as an input because its state remains constant and unaffected by other variables. Inputs, which often denote external stimuli [4], are explicitly set by the modeler to perform experiment campaigns. Conversely, *IRAK4* and *PIK3AP1* can be considered output variables because they do not appear in the update functions of other variables.

Step 1: initial partition. Our reduction algorithm starts with the specification of an initial partition of variables. The idea behind initial partitions is that the modeler can force our algorithm to not collapse given variables, by placing them in different initial blocks. In the case studies presented in the *Results and discussion* section we see examples of user-specified initial partitions enabling analyses of interest on the

$$
\begin{aligned}
x_{TLR5}(t+1) &= f_{x_{TLR5}} = x_{TLR5}(t) \\
x_{TICAM1}(t+1) &= f_{x_{TICAM1}} = x_{TLR5}(t) \\
x_{MyD88}(t+1) &= f_{x_{MyD88}} = x_{TLR5}(t) \\
x_{IRAK4}(t+1) &= f_{x_{IRAK4}} = x_{MyD88}(t) \vee x_{TICAM1}(t) \\
x_{PIK3AP1}(t+1) &= f_{x_{PIK3AP1}} = x_{MyD88}(t)
\end{aligned}
$$

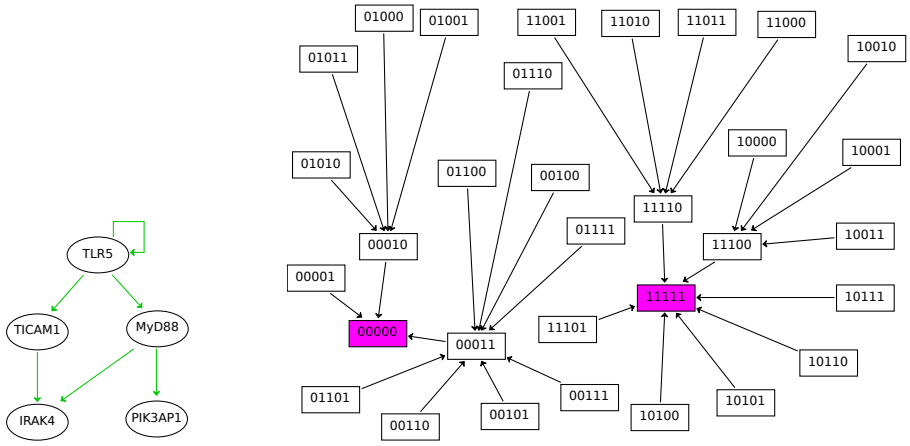


Figure 2: Excerpt of the BN from [30]. It refers to the receptor *TLR5* and its signalling to the four following genes: *TICAM1, MyD88, IRAK4, PIK3AP1*. When a virus infects an organism, the receptor *TLR5* receives the relevant antigen stimuli becoming active (the value of x_{TLR5} turns from 0 to 1), and the signal is subsequently propagated to the other connected genes. (Top) The update functions of the BN. (Bottom-left) Variables are commonly depicted as nodes in a network while directed links represent influences between them. A directed link from a source variable to a target variable denotes that the source variable exists in the update function of the target variable. (Bottom-right) The corresponding STG, where we use purple to denote attractors.

considered models. This is how initial partitions shall be used, devising case-by-case useful ones. In order to favour a systematic large-scale validation of our approach, here we consider two examples of initial partitions whose computation can be easily automated: the *maximal* partition, where all variables are placed in the same block; and the *input-separated* (IS) one, where the inputs are separated from the other variables, i.e., we use an initial partition with two blocks, one for input variables and one for the other variables. [1] In the example, these are respectively given by

[1] We refer to [34] for a third example of initial partition, *input-distinguished*, where inputs were further separated from each other. As exemplified in the case studies in the *Results and discussion* section, initial partitions should be defined by the modeler depending on the model at hand and on the properties to be studied. We discuss the maximal and IS partitions here to enable the large-scale validation of BBE discussed in the same section.

the partitions:

$$\mathcal{H}_0 = \big\{\{x_{TLR5}, x_{TICAM1}, x_{MyD88}, x_{IRAK4}, x_{PIK3AP1}\}\big\} \tag{1}$$

and

$$\mathcal{H}'_0 = \big\{\{x_{TLR5}\}, \{x_{TICAM1}, x_{MyD88}, x_{IRAK4}, x_{PIK3AP1}\}\big\}. \tag{2}$$

Iterative step: splitting by the BBE condition. At every iteration, the algorithm checks the BBE condition on the current partition. Formally, BBE is defined as a partition X of variables that renders the following formula valid:

$$\Phi^{\mathcal{H}} \equiv \left(\bigwedge_{\substack{H_i \in \mathcal{H} \\ x,x' \in H_i}} \Big(x = x'\Big) \right) \longrightarrow \bigwedge_{\substack{H_i \in \mathcal{H} \\ x,x' \in H_i}} \Big(f_x = f_{x'}\Big) \tag{3}$$

This is a Boolean formula for: *whenever all variables in the same block have same value, they will not be distinguished in the next state.* In other words, $\Phi^{\mathcal{H}}$ says that if for all partition blocks H_i the variables in H_i are equal, then the evaluations of update functions of variables in the same block stay equal. A SAT solver can determine if $\Phi^{\mathcal{H}}$ is valid by checking the unsatisfiability of its negation. For example, given the $\mathcal{H}'_0$ partition in Eq. 2, one can obtain that $\neg\Phi^{\mathcal{H}'_0}$ is satisfiable (i.e., $\mathcal{H}'_0$ is not a BBE) because there exists the assignment s given by

$$s = (s_{x_{TLR5}}, s_{x_{TICAM1}}, s_{x_{MyD88}}, s_{x_{IRAK4}}, s_{x_{PIK3AP1}}) = (1, 0, 0, 0, 0)$$

for which, as it can be seen in the STG of Fig. 2, the next state s' is

$$s' = (s'_{x_{TLR5}}, s'_{x_{TICAM1}}, s'_{x_{MyD88}}, s'_{x_{IRAK4}}, s'_{x_{PIK3AP1}}) = (1, 1, 1, 0, 0).$$

This assignment proves that variables x_{TICAM1}, x_{MyD88}, x_{IRAK4}, and $x_{PIK3AP1}$ cannot belong to the same block of a partition that satisfies the BBE criteria because despite having the same value (0) in the source state s, they differ in the target state s'. In addition, the assignment s suggests to split that block into two sub-blocks for which that assignment does not disprove the BBE condition: x_{TICAM1} and x_{MyD88} have same value in s', as well as x_{IRAK4} and $x_{PIK3AP1}$. Thus the algorithm will perform a new iteration with the refined partition

$$\mathcal{H}'_1 = \big\{\{x_{TLR5}\}, \{x_{TICAM1}, x_{MyD88}\}, \{x_{IRAK4}, x_{PIK3AP1}\}\big\}.$$

With this, $\neg\Phi^{\mathcal{H}'_1}$ is unsatisfiable, implying that $\mathcal{H}'_1$ is a BBE partition. In Theorem 2 from Additional file 1, we prove that this algorithm returns, for any initial partition, its unique coarsest refinement that satisfies the BBE condition (3). Overall, the algorithm takes at most n steps, where n is the number of BN variables; at every step, it iterates through the provided SAT assignment, if any is provided, to perform the splitting. Thus, overall the algorithm is as hard as SAT solving; however, the numerical evaluation presented in the *Results and discussion* section will show how it can effectively tackle BN models from the literature.

BBE properties. As discussed in Fig. 1, given a BBE it is possible to construct a reduced BN where each variable represents a partition block (Proposition 4 from Additional file 1). The STG of the reduced BN agrees with the original STG on all, and only on, states that are *constant* on the partition, i.e., whose variables in the same block have the same value (Proposition 4 from Additional file 1). The reduction also preserves any attractor of the original BN which contains at least one state that is constant on the partition (Theorem 5 from Additional file 1). Thus, in particular the reduced BN maintains the exact length of the attractors that are preserved without introducing spurious dynamical behavior. Instead, all states non constant on the partition are dropped, as well as all attractors not containing any state constant on the BBE partition.

We use two examples to better explain the exact preservation of part of the attractors. Considering preserved attractors, we have seen in Fig. 1 that the two-states attractor of the original model (Fig. 1 bottom-left) is preserved in a two-states attractor in the reduced model (Fig. 1 bottom-right). This is the case for any preserved attractor; the number of states is preserved.

As regards attractors that are not preserved, we provide in Fig. 3 (top-left) a simple BN with 3 variables (x_1, x_2, and x_3) and 4 attractors (steady-states, Fig. 3, bottom-left). Fig. 3 (top-right) shows a BBE reduction of the model where x_1 and x_2 get collapsed. We can see in Fig. 3 (bottom-right) that 2 attractors are preserved in the BBE reduction, while the other 2 attractors belong to the part of the STG that is not preserved, and therefore are not present in the reduced BN. In particular, according to our theory, the two attractors where x_1 and x_2 are both 1 or both 0 are preserved. Instead, the other two attractors have different values for x_1 and x_2, and therefore are not preserved.

Partially asynchronous BNs. In Additional file 1, we show how BBE can also be applied to partially asynchronous BNs. Here, we equip a BN with a partition $\mathcal{K}$ of its variables that we name *synchronization partition*. A new state is obtained by selecting one of the blocks K of $\mathcal{K}$, and then applying of the update functions of the variables in K only. The activation values of the other variables are not modified. Notably, this synchronization schema is supported, e.g., by popular BN analysis tools like GINsim [6] under the name of *priority classes* [7].[2] In particular, BBE can be applied to such BNs with the caveat that the initial partition must be $\mathcal{K}$ or refinements of it. In Additional file 3 we apply BBE to a BN with partially asynchronous update schema.

Results and discussion

In this section, we perform a large-scale validation of BBE. We first check its reduction power on published models from the literature, and then we demonstrate how it facilitates the analysis tasks of STG generation and attractor computation. In particular, we show how BBE brings important analysis speed-ups, both in terms of

[2] The dynamics of BNs considered in this paper are less general than those offered by GINsim using priority classes, as they also further allow to assign different priorities to the classes, and to update the variables within them asynchronously.

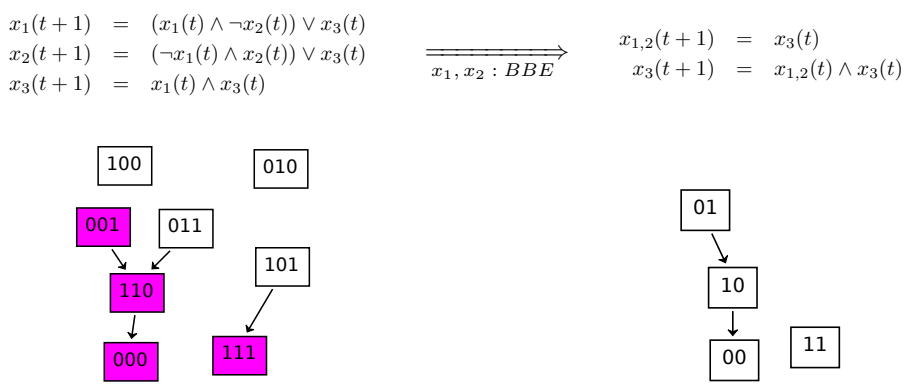


Figure 3: Boolean backward equivalence shown on a simple example: not all attractors are preserved. (Top-left) BN with three variables denoted by x_1, x_2, and x_3. (Bottom-left) The underlying STG. The model has 4 steady-state attractors (nodes `100`, `010`, `000`, and `111`). Two have same activation values for x_1 and x_2 (`000`, and `111`), two have not. (Top-right) Variables x_1 and x_2 can be shown to be BBE-equivalent by inspecting their update functions. If they have the same value in a state, i.e. $x_1(t) = x_2(t)$, then they will be equivalent for all successor states. Based on this, a reduced BN can be obtained by considering a representative variable for each block and rewriting the corresponding update functions in terms of those representatives (here the representative variable is denoted by $x_{1,2}$). (Bottom-right) The underlying STG agrees with the original one on all states that have equal values for variables in the same block (purple nodes in bottom-left panel). Notably, the two attractors having same activation value for x_1 and x_2 are preserved, while the other two are dropped, as expected by our theory.

STG generation, and attractor analysis. In several cases, BBE enables the analysis of models that were originally intractable due to their complexity.

After this, we use two selected case studies to show how one can tune the reduction power of BBE to preserve or exclude specific dynamics of interest.

Toolchain. We implemented our method in ERODE [32], a freely available software for the modeling, analysis, and reduction of biological systems modelled in terms of BNs [34], differential equations [35], and chemical reaction networks [19]. ERODE integrates the SAT solver Z3 [36]. Thanks to importing/exporting functionalities, we let ERODE interact with the COLOMOTO Notebook [37, 38], which integrates several tools for modeling and analysis of BNs. STG generation is performed using the tool PyBoolNet [39], while attractor identification is performed using the SAT-based tool BNS [12].[3]

[3] Among the tools available in the COLOMOTO notebook, we could have opted for GINsim [40] and BoolSim [41] for STG generation and attractor identification, respectively, for BNs with synchronous update schema. In both cases, we have

Configuration. All experiments were conducted on a machine equipped with an Intel Xeon(R) 2.80 GHz processor and 32 GB RAM. We imposed an arbitrary timeout of 8 hours for each task, after which we terminated the analysis. We refer to these cases as *time-out*, while we use *out-of-memory* if the execution issued a memory error.

We conducted our investigation using two model repositories: GINsim repository [6] (`http://ginsim.org/models_repository`), which contains 83 models, and the Biomodels repository [42] (`https://www.ebi.ac.uk/biomodels/`), which contains 24 models, obtaining overall 98 distinct models (9 appeared in both repositories). From these, we restricted only to models with input variables, obtaining 86 models. In other words, we considered about 92% of the models available in the two respositories. This selection was done to avoid favouring BBE: in BNs without inputs, IS initial partitions correspond to the maximal ones, which, as the name says, allow for the best possible BBE reduction of a model in terms of aggregation power. Part of these 86 models, 45, are multi-valued networks, i.e. logical models wherein some variables take more than two activation statuses, e.g., $\{0, 1, 2\}$ for *low*, *medium*, or *high* concentration respectively (see, e.g., [43]). We transform such models in dynamically equivalent BNs by applying a so-called *booleanization* technique [44], supported by GINsim [40].

As in the *Methods* section, we consider two reduction scenarios relevant to input variables, using *maximal* and *input-separated* (IS) initial partitions like $\mathcal{H}_0$ in Eq. (1) and $\mathcal{H}'_0$ in Eq. (2), respectively. In Additional file 4 we perform a similar analysis on randomly generated BNs.

Large-scale validation

Large-scale validation: reduction power. We begin by addressing the reduction power of BBE. For this, we consider the *reduction ratios* (variables in the reduced BN over the variables in the original one) obtained on all models.

Fig. 4 displays the reduction ratios for both the maximal and IS reductions. We observe that almost all models can be reduced by BBE, in particular 93% admit a maximal reduction, while 91% admit an IS one. The reduction ratios distribute almost uniformly from 0.15 to 1.00 (no reduction), with average reduction ratio of 0.70 and 0.77 for maximal and IS reductions, respectively. For most models, the maximal and IS reduction ratios do not change significantly, meaning that BBE is effective also when we prevent input variables from merging with internal variables. All detailed results of this analysis can be found in Table S3 in Additional file 5.

Large-scale validation: STG generation speed-up. We hereby demonstrate the speed-ups that BBE provides to STG generation on a selection of the considered models. Fig. 5 focuses on the 20 models with more than 10 variables for which the STG generation succeeded in both the original and reduced models, while Fig. 6 focuses on the 13 ones where the STG generation failed on the original model and succeeded in the maximal or IS reduction. On the used machine, STG generation

opted for the tools with best performances according to preliminary experiments we conducted, not reported here.

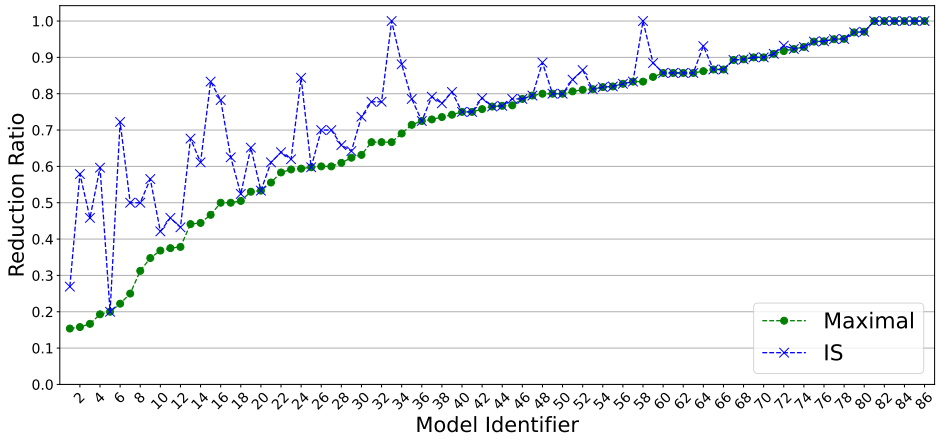


Figure 4: Large-scale validation: reduction power. The x-axis provides model identifiers for the 86 considered models (only even ones are shown due to space constraints), while the y-axis refers to reduction ratios "reduced variables over original ones". The green dots denote the reduction ratios, in increasing order, for maximal reductions. Using the same ordering, the blue crosses denote the reduction ratios for IS reductions. Only 6 models do not admit any BBE reduction, while two more models (33 and 58) do not admit IS reduction.

failed for models with 24 or more variables. Therefore, Fig. 5 focuses on models with less than 24 variables, while Fig. 6 focuses on models with 24 or more variables for which at least one reduction had less than 24 variables.

The red line in Fig. 5 marks the area where the reduction would not bring speed-ups. We can see that all points are below the line, with instances showing more than two orders of magnitude difference between the original and reduced runtimes. This proves that BBE can effectively lead to faster STG generation. All cases where the dots and crosses overlap refer to models where the two reductions coincide.

We now consider the 13 models in Fig. 6 where STG generation was not feasible for the original models. We note that the generation succeeded for all maximal reductions, while it failed for two IS ones. As denoted by the model identifiers in the x-axis, these are models 4 and 15 in Fig. 4, where the IS reductions have 34 and 25 variables, respectively. The largest runtime is 441 s for the IS reductions, and 338 s for the maximal ones. Detailed results are presented in Table S4 in Additional file 5.

Large-scale validation: Attractor computation speed-up. Fig. 7 studies the speed-ups that BBE provided to the computation of attractors on the models from Fig. 4. The plot has the same structure as Fig. 5. We observe that, in several cases, we have significant analysis speed-ups. In particular, we note how the dots and crosses spread to the right, due to original runtimes in the order of 10^3 s, while they hardly go up beyond 1 s for runtimes on reduced models. Furthermore, models 18 and 29 from Fig. 4 are omitted here because the analysis failed on the original models. Instead, the analysis of their maximal reductions required at most 0.15 s, and that of their IS reductions at most 2.5 s. Detailed results are given in Table S5 in Additional file 5.

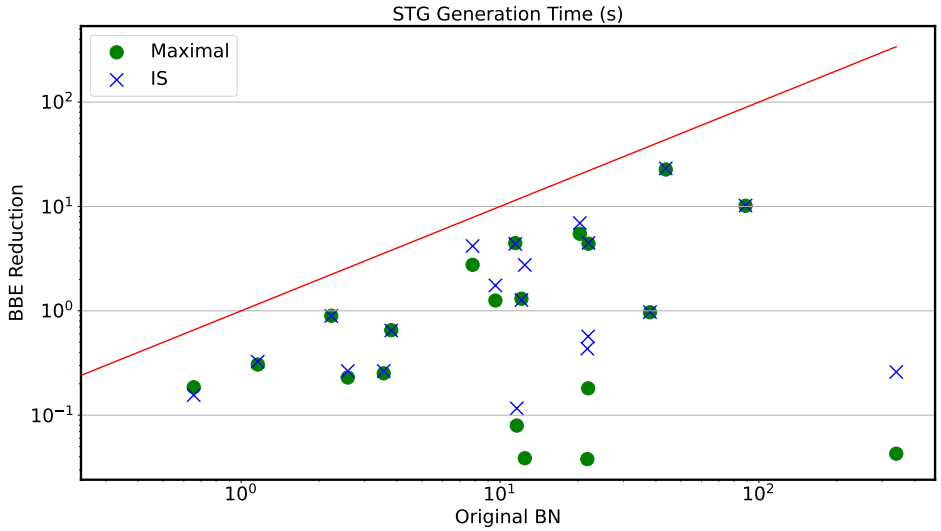


Figure 5: Large-scale validation: STG generation speed-up. Comparison of STG generation between BBE reductions and original BNs. We omit models with 10 or fewer variables, where the runtimes are not particularly informative because STG generation is trivial. Furthermore, we omit models with more than 60 variables, where STG generation fails with *out-of-memory* for both the original models and their reductions. Overall, we obtain 33 models, from which here we focus only on the 20 ones for which the STG generation succeeded in both the original and reduced models, while Fig. 6 focuses on the remaining 13. The x-axis refers to the generation time for original models, while the y-axis refers to that for reduced models, using green circles and blue crosses for maximal and IS reductions, respectively. The runtimes are averaged over 3 runs.

Large-scale validation: Interpretation. BBE can successfully reduce a large amount of models. For the original models, the state space explosion prevents full state space exploration in many cases, and hampers the identification of the attractors. This is mitigated in practice by BBE, with extreme cases where BBE made analysis feasible whereas the original models were intractable. As shown in Table S5 in Additional file 5, part of the attractors are lost in the reduced models, namely those not involving constant states on the computed BBE (see *Methods* section). Table S5 shows cases like model 18 or 29, whose attractors could not be computed at all without BBE reduction. At the same time, the table shows that, by using the *default* IS or maximal initial partitions, a large part of the attractors might be lost (because, according to our theory, involve STG states where variables belonging to the same BBE block have different value). In Fig. S7 from Additional file 4 we provide this information graphically for the BNs from the two repositories, and for randomly generated ones. This can be mitigated by devising *refined* initial partitions for the model and problem at hand. This is exemplified and discussed in greater detail in the next section where we show how a modeler can easily devise refined initial partitions that may allow to preserve more attractors, or drop those that are not of interest.

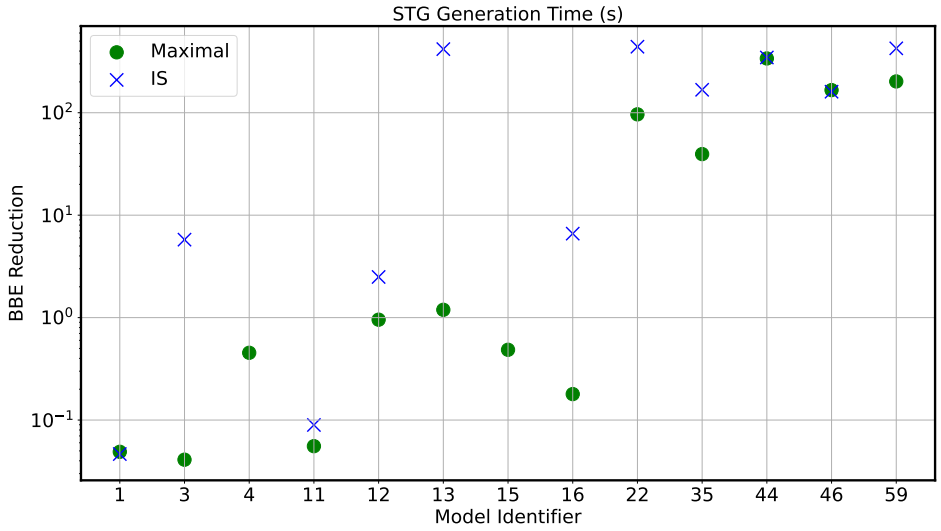


Figure 6: Large-scale validation: STG generation speed-up. STG generation time for the maximal and the IS reductions. We consider the 13 models omitted in Fig. 5 because the STG generation failed for the original models. The x-axis refers to the model identifier from Fig. 4, while the y-axis refers to the generation time for the reduced models, using green circles and blue crosses for maximal and IS reductions, respectively. The runtimes are averaged over 3 runs.

Case Studies

In the previous part of this section we studied the aggregation power and the analysis speedups offered by BBE on 86 models from the literature. Here, instead, we use two selected case studies (MAPK, T-LGL) to show how one can tune, or *refine* the reduction power of BBE using model-specific information to preserve all dynamics of interest, and selectively exclude behavior that does not have biological relevance. Nevertheless, for completeness, we provide in Table 1 information on the analysis runtimes on all models (and their reductions) discussed in this section. We consider analysis runtimes on the models (*Original*), and on their IS (*IS*) and maximal (*Maximal*) reductions as done in the large-scale validation. Furthermore, we consider an additional reduction obtained using a refined initial partition discussed in the corresponding sections (*Refined*). STG generation failed on all models and reductions because they all have more than 24 variables. Indeed, we have previously discussed how, on the used machine, STG generation fails for models with 24 or more variables. Instead, attractor analysis succeeded on all models, with important speed-ups obtained for all reductions. For both models, the *IS* and *Maximal* cases have a particularly low analysis runtime. This is because, as we shall discuss next, several attractors are discarded in these reductions. Notably, despite the *Refined* reductions have speedup factors of about two, as we shall see they preserve all attractors for MAPK, and all attractors *of interest* for T-LGL.

MAPK case study. We consider a BN model for Mitogen-Activated Protein Kinase (MAPK) from [45]. The model consists of tightly interconnected signalling

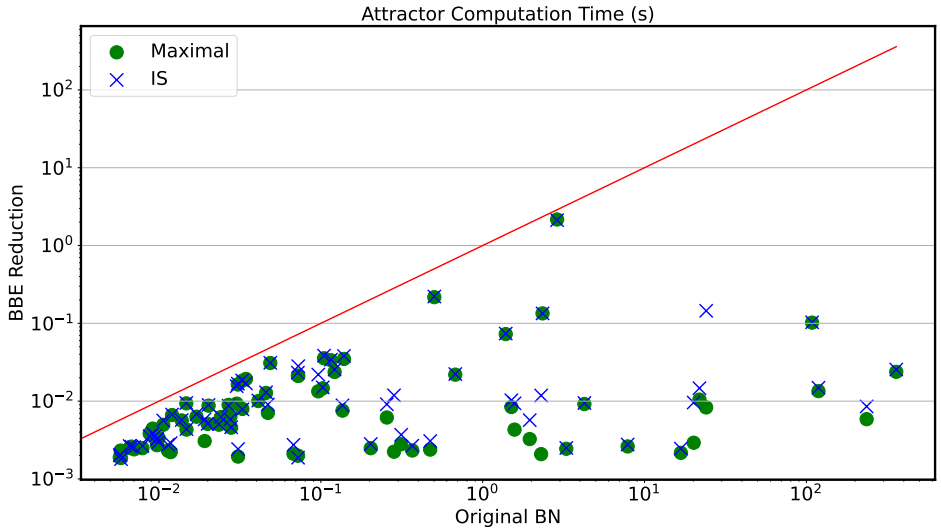


Figure 7: Large-scale validation: Attractor computation speed-up. Attractor computation time of the original models versus the one of maximal and IS reductions. Out of the 86 models from Fig. 4 we select the 78 admitting both maximal and IS reduction. The figure further omits models 18 and 29 from Fig. 4 for which the analysis failed for the original model due to *time-out*. The x-axis refers to the analysis time for original models, while the y-axis refers to that for reduced models, using green circles and blue crosses for maximal and IS reductions, respectively. The runtimes are averaged over 3 runs.

	MAPK				T-LGL			
	Original	*IS*	*Maximal*	*Refined*	*Original*	*IS*	*Maximal*	*Refined*
STG Generation	out-of-memory				out-of-memory			
Atractor analysis	0.55	0.16	0.16	0.35	2.66	0.10	0.11	1.17

Table 1: Analysis runtimes (in seconds) for the models in section Case Studies.

pathways involved in diverse cellular processes, such as cell cycle, survival, apoptosis and differentiation. The BN is depicted in Fig. 8. It contains 53 variables, 4 of which being inputs (*EGFR_stimulus*, *FGFR3_stimulus*, *TGFBR_stimulus*, and *DNA_damage*), and has 40 attractors.

MAPK: Maximal and IS reduction. The maximal BBE reduction of this model has 39 variables. The discovered blocks are visualized in Fig. 8 using different background colors. In particular, we note that the yellow block contains all inputs and five non-inputs variables, three related to *TGFBR_stimulus*, and two related to *DNA_damage*. Instead, the IS reduction has 41 variables, the only difference being that the block with inputs from the maximal reduction (Fig. 8) is split in three blocks: one for the inputs, one for the two non-input variables directly connected to the two right-most inputs, and one for the remaining non-input variables. In both cases, the reduced BNs have 17 attractors.

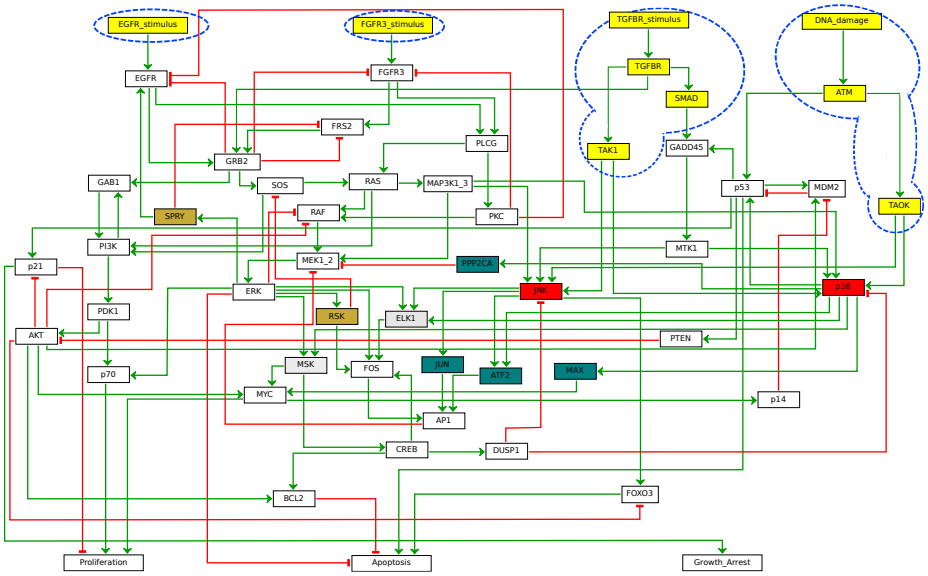


Figure 8: Graphical representation of the MAPK BN using GINsim. The background colors denote blocks of the maximal BBE (white background denotes singleton blocks). Instead, the blue dashed shapes denote blocks of the refined initial partition, vertical IS, where we omit the fifth large block containing all remaining nodes.

MAPK: Refined reduction with vertical IS. We propose a third model-specific initial partition that considers inputs also *indirectly*. Intuitively, variables like *TGFBR* depend only on the value assigned to an input (*TGFBR_stimulus*). This reasoning can be iterated downward through the pathway, allowing to add also *TAK1*, and *SMAD*, until variables that depend on other (input) variables are met. In some sense, we can see *TGFBR*, *TAK1*, and *SMAD* as indirect inputs. This is because, in a few iterations the value assigned to the corresponding input will be propagated to them, and they will not change value anymore. In other words, we use a block per input, each containing the input and all non-input variables only positively affected by the input or by variables in the block. That way, we obtain an initial partition denoted by the blue dashed shapes in Fig. 8, plus an additional fifth block containing all other variables. The rationale is that a variable only affected by an input will have the same truth value of the input, therefore it can be considered as a sort of indirect input. The obtained BBE is depicted in Fig. 9. The reduced BN contains 42 variables and preserves all 40 attractors.

T-LGL case study. We consider a BN model for T-LGL from [20]. It refers to the disease T-LGL leukemia which features a clonal expansion of antigen-primed, competent, cytotoxic T lymphocytes (CTL). This BN is a signalling pathway, constructed empirically through extensive literature review, and determines the survival of CTL. The BN, depicted in Fig. 10, consists of 60 variables, 6 of which are inputs (the yellow nodes in Fig. 10). The model has 264 attractors.

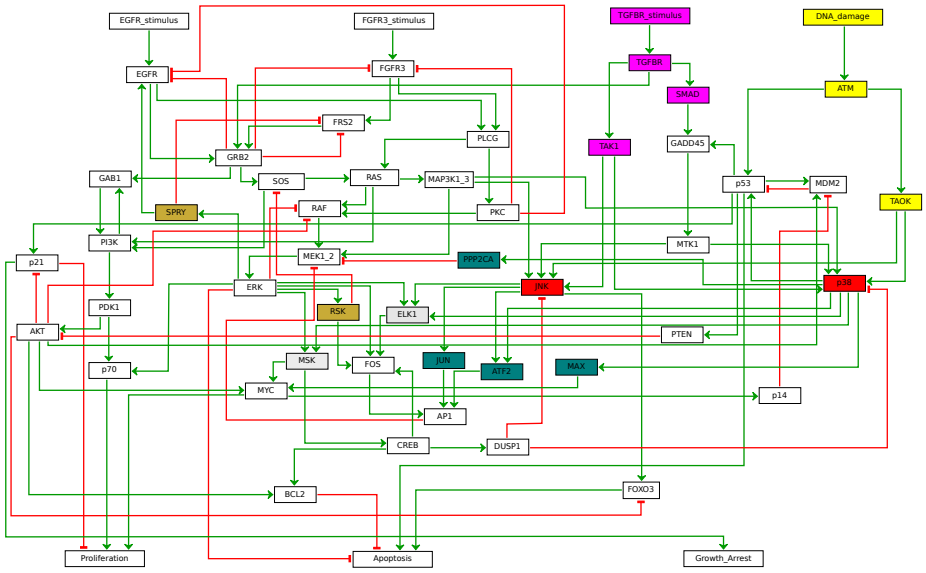


Figure 9: Graphical representation of the MAPK BN. Background colors denote blocks of the BBE obtained using the refined initial partition (white background denotes singleton blocks).

T-LGL: Maximal and IS reduction. The maximal and IS BBE coincide, as depicted in Fig. 10. We have only two non-singleton blocks: one consisting of all the inputs, and one consisting of *FasT*, *A20*, *TNF*, and *RANTES*. The reduced BN has 52 variables and 6 attractors, which means that most of the attractors are lost.

T-LGL: Refined reduction. In [20], the authors discover that the simultaneous activation of the two input variables *IL15* and *PDGF* is sufficient to produce all dynamics of interest to them (namely, all the known so-called deregulations and signalling abnormalities).

In terms of initial partitions for BBE, we can encode the notion of *contemporary activation or deactivation* of the two inputs by using a model- and problem-specific initial partition where *IL15* and *PDGF* form a block. Furthermore, we assign every input to a singleton block, while all non-input variables belong to the same block, for a total of 56 blocks. It turns out that this initial partition is actually a BBE, which therefore does not get refined by our algorithm. The reduced BN has 120 attractors.

Conclusion

Boolean backward equivalence (BBE) is an automatic reduction technique for Boolean networks (BNs) which exactly preserves dynamics of interest to the modeler by collapsing variables that are proven to have equal values in all states. The method, based on a partition refinement algorithm, can be tuned on a model- and problem-specific way by specifying which variables should be preserved using an appropriate choice of the initial partition. The approach is complementary to the state of the art. Roughly, in [4, 13], reduction is achieved by replacing variables with constants and propagating those in the transitions of somehow richer STGs or across

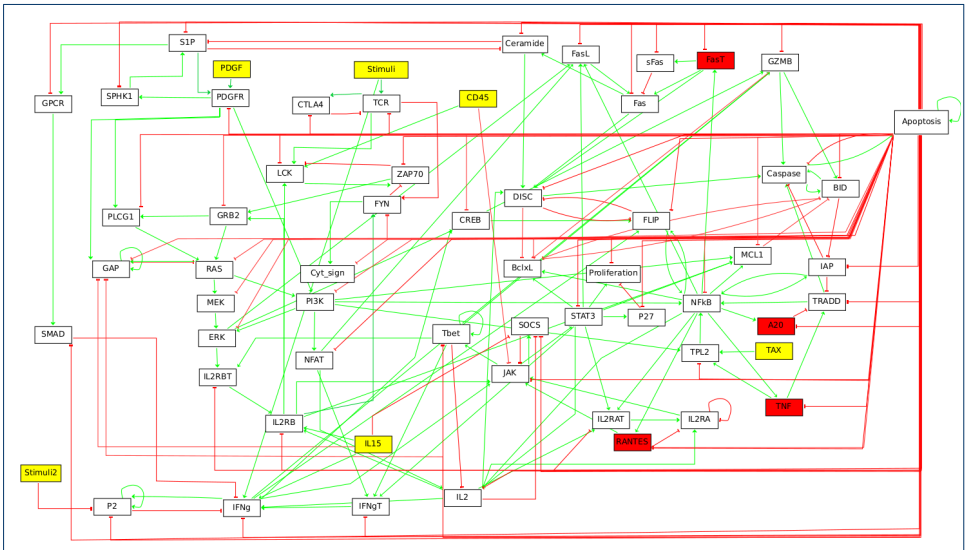


Figure 10: Graphical representation of the T-LGL BN using GINsim. Background colors denote blocks of both the maximal and IS BBE, which coincide (white background denotes singleton blocks).

the network, respectively. Thus, the reduced model cannot be used to investigate how changes in those variables affect the dynamics. In a BBE reduction, instead, variables are collapsed into blocks and the original dynamics is exactly recovered whenever variables in the same block are assigned equal values. These studies [4, 13] additionally remove the output [4] variables (also called leaf variables [13]). However, output variables sometimes are used to denote different "responses" by the modelled system [4, 30], therefore their removal might not always be appropriate.

In variable absorption [14, 15], the main assumption is that there are variables that are updated faster than others, therefore one class of variables can be assumed to be constant and absorbed if focusing on the dynamics of the other class. Unlike BBE, this can only increase the number of attractors. In particular, variable absorption preserves exactly all steady states (single-state attractors), while it might change the length of other attractors. Furthermore, new spurious attractors might be added. Instead, BBE might decrease the number of attractors (it discards all and only the attractors involving states where BBE-equivalent variables have different activation values), but all preserved attractors are preserved exactly, including their length and reachibility from (preserved) initial states, and no spurious ones are added. Regarding other relevant work, in [16], the authors identify variables that have the same value in attractors only, but, differently from BBE, might behave differently in other states of the STG.

We validated BBE on 86 BNs from two model repositories, providing reductions and analysis speed-ups in almost all cases. In some, BBE enabled the analysis of models which would be otherwise intractable. There were also instances for which the reduced model could not be analyzed. This calls for further research into more aggressive reductions; for example, in its current implementation multi-valued BNs are first translated into ordinary BNs, but this causes a blow-up in the number of variables. It is worth investigating approaches that circumvent the intermediate

translation to reduce dimensionality. Another area of research concerns the different semantic interpretations of a BN. Currently, BBE supports BN with synchronous and partially asynchronous updates; we plan to investigate variants of BBE for probabilistic BNs.

Supplementary Information

Declarations

Abbreviations

BN: Boolean Network. BBE: Boolean Backward Equivalence. STG: State Transition Graph.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Availability of data and materials

Material to replicate the large-scale experiments, including all models, are available here: https://www.erode.eu/models/BMCBioInf_CMSB2021.zip

Competing interests

The authors declare that they have no competing interests.

Funding

Partially supported by the DFF project REDUCTO 9040-00224B, the Poul Due Jensen Grant 883901, the Villum Investigator Grant S4OS, and the PRIN project SEDUCE 2017TWRCNB.

Authors' contributions

All authors contributed equally and read and approved the final manuscript.

Acknowledgements

We thank reviewers of the original CMSB 2021 submission and of this submission for their help in improving the paper. We also thank the CMSB 2021 attendants for their comments and suggestions on the work. We thank Laure Bally-Cuif, author of [46], for her fruitful information on the modelling approach helping us in the case study in Additional file 3

Additional Files

Additional file 1 — Technical Results.
Additional file 2 — Comparison with encoding-based reductions.
Additional file 3 — Application of BBE to a partially asynchronous schema
Additional file 4 — An application of BBE to random Boolean Networks
Additional file 5 — Tables with detailed results from large-scale experiments.

Author details
[1]Department of Applied Mathematics and Computer Science, Technical University of Denmark, Lyngby, Denmark. [2]SysMA Unit, IMT School for Advanced Studies, Lucca, Italy. [3]Department of Computer Science, University of Aalborg, Denmark. [4]Department of Excellence EMbeDS and Institute of Economics, Sant'Anna School for Advanced Studies, Pisa, Italy.

References

1. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology **22**(3), 437–467 (1969)
2. Wang, R.-S., Saadatpour, A., Albert, R.: Boolean modeling in systems biology: an overview of methodology and applications. Physical biology **9**(5), 055001 (2012)
3. Azpeitia, E., Benítez, M., Vega, I., Villarreal, C., Alvarez-Buylla, E.R.: Single-cell and coupled grn models of cell patterning in the arabidopsis thaliana root stem cell niche. BMC systems biology **4**(1), 1–19 (2010)
4. Naldi, A., Monteiro, P.T., Chaouiya, C.: Efficient handling of large signalling-regulatory networks by focusing on their core control. In: International Conference on Computational Methods in Systems Biology, pp. 288–306 (2012). Springer
5. Bérenguier, D., Chaouiya, C., Monteiro, P.T., Naldi, A., Remy, E., Thieffry, D., Tichit, L.: Dynamical modeling and analysis of large cellular regulatory networks. Chaos: An Interdisciplinary Journal of Nonlinear Science **23**(2), 025114 (2013)
6. Naldi, A., Berenguier, D., Fauré, A., Lopez, F., Thieffry, D., Chaouiya, C.: Logical modelling of regulatory networks with ginsim 2.3. Biosystems **97**(2), 134–139 (2009)
7. Fauré, A., Naldi, A., Chaouiya, C., Thieffry, D.: Dynamical analysis of a generic boolean model for the control of the mammalian cell cycle. Bioinformatics **22**(14), 124–131 (2006)
8. Schwab, J.D., Kühlwein, S.D., Ikonomi, N., Kühl, M., Kestler, H.A.: Concepts in boolean network modeling: What do they all mean? Computational and Structural Biotechnology Journal **18**, 571–582 (2020). doi:10.1016/j.csbj.2020.03.001
9. Hopfensitz, M., Müssel, C., Maucher, M., Kestler, H.A.: Attractors in boolean networks: a tutorial. Computational Statistics **28**(1), 19–36 (2013)
10. Drossel, B.: Random boolean networks. Reviews of nonlinear dynamics and complexity **1**, 69–110 (2008)
11. Akutsu, T., Kuhara, S., Maruyama, O., Miyano, S.: A system for identifying genetic networks from gene expression patterns produced by gene disruptions and overexpressions. Genome Informatics **9**, 151–160 (1998)
12. Dubrova, E., Teslenko, M.: A sat-based algorithm for finding attractors in synchronous boolean networks. IEEE/ACM transactions on computational biology and bioinformatics **8**(5), 1393–1399 (2011)
13. Bilke, S., Sjunnesson, F.: Stability of the Kauffman model. Physical Review E **65**(1), 016129 (2001)
14. Naldi, A., Remy, E., Thieffry, D., Chaouiya, C.: Dynamically consistent reduction of logical regulatory graphs. Theoretical Computer Science **412**(21), 2207–2218 (2011)
15. Veliz-Cuba, A.: Reduction of boolean network models. Journal of theoretical biology **289**, 167–172 (2011)
16. Richardson, K.A.: Simplifying boolean networks. Advances in Complex Systems **8**(04), 365–381 (2005)
17. Figueiredo, D.: Relating bisimulations with attractors in boolean network models. In: International Conference on Algorithms for Computational Biology, pp. 17–25 (2016). Springer
18. Zañudo, J.G.T., Albert, R.: An effective network reduction approach to find the dynamical repertoire of discrete dynamic networks. Chaos: An Interdisciplinary Journal of Nonlinear Science **23**(2), 025111 (2013). doi:10.1063/1.4809777
19. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Maximal aggregation of polynomial dynamical systems. Proceedings of the National Academy of Sciences **114**(38), 10029–10034 (2017)
20. Zhang, R., Shah, M.V., Yang, J., Nyland, S.B., Liu, X., Yun, J.K., Albert, R., Loughran, T.P.: Network model of survival signaling in large granular lymphocyte leukemia. Proceedings of the National Academy of Sciences **105**(42), 16308–16313 (2008)
21. Sproston, J., Donatelli, S.: Backward bisimulation in markov chain model checking. Software Engineering, IEEE Transactions on **32**(8), 531–546 (2006). doi:10.1109/TSE.2006.74
22. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Forward and backward bisimulations for chemical reaction networks. In: 26th International Conference on Concurrency Theory, CONCUR 2015, Madrid, Spain, September 1.4, 2015, pp. 226–239 (2015). doi:10.4230/LIPIcs.CONCUR.2015.226. https://doi.org/10.4230/LIPIcs.CONCUR.2015.226
23. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Symbolic computation of differential equivalences. ACM SIGPLAN Notices **51**(1), 137–150 (2016)
24. Tognazzi, S., Tribastone, M., Tschaikowski, M., Vandin, A.: Differential equivalence for linear differential algebraic equations. IEEE Trans. Autom. Control. **67**(7), 3484–3493 (2022). doi:10.1109/TAC.2021.3108530
25. Tognazzi, S., Tribastone, M., Tschaikowski, M., Vandin, A.: Backward invariance for linear differential algebraic equations. In: 57th IEEE Conference on Decision and Control, CDC 2018, Miami, FL, USA, December 17-19, 2018, pp. 3771–3776. IEEE, ??? (2018). doi:10.1109/CDC.2018.8619710. https://doi.org/10.1109/CDC.2018.8619710
26. Cardelli, L.: Morphisms of reaction networks that couple structure to function. BMC systems biology **8**(1), 84 (2014)
27. Biere, A., Biere, A., Heule, M., van Maaren, H., Walsh, T.: Handbook of Satisfiability: Volume 185 Frontiers in Artificial Intelligence and Applications. IOS Press, NLD (2009)
28. Valmari, A., Franceschinis, G.: Simple $O(m \log n)$ time markov chain lumping. In: Tools and Algorithms for the Construction and Analysis of Systems, 16th International Conference, TACAS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings, pp. 38–52 (2010). doi:10.1007/978-3-642-12002-2_4. http://dx.doi.org/10.1007/978-3-642-12002-2_4
29. Paige, R., Tarjan, R.: Three partition refinement algorithms. SIAM Journal on Computing **16**(6), 973–989 (1987). doi:10.1137/0216062. http://epubs.siam.org/doi/pdf/10.1137/0216062
30. Rodríguez-Jorge, O., Kempis-Calanis, L.A., Abou-Jaoudé, W., Gutiérrez-Reyna, D.Y., Hernandez, C., Ramirez-Pliego, O., Thomas-Chollier, M., Spicuglia, S., Santana, M.A., Thieffry, D.: Cooperation between t cell receptor and toll-like receptor 5 signaling for cd4+ t cell activation. Science signaling **12**(577) (2019)
31. Klamt, S., Saez-Rodriguez, J., Lindquist, J.A., Simeoni, L., Gilles, E.D.: A methodology for the structural and

functional analysis of signaling and regulatory networks. BMC bioinformatics **7**(1), 56 (2006)
32. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Erode: a tool for the evaluation and reduction of ordinary differential equations. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems, pp. 310–328 (2017). Springer
33. Le Novere, N., Bornstein, B., Broicher, A., Courtot, M., Donizelli, M., Dharuri, H., Li, L., Sauro, H., Schilstra, M., Shapiro, B., *et al.*: Biomodels database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. Nucleic acids research **34**(suppl_1), 689–691 (2006)
34. Argyris, G., Lluch Lafuente, A., Tribastone, M., Tschaikowski, M., Vandin, A.: Reducing boolean networks with backward boolean equivalence. In: International Conference on Computational Methods in Systems Biology, pp. 1–18 (2021). Springer
35. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Symbolic computation of differential equivalences. In: POPL 2016, pp. 137–150 (2016). doi:10.1145/2837614.2837649
36. De Moura, L., Bjørner, N.: Z3: An efficient smt solver. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems, pp. 337–340 (2008). Springer
37. Naldi, A., Hernandez, C., Levy, N., Stoll, G., Monteiro, P.T., Chaouiya, C., Helikar, T., Zinovyev, A., Calzone, L., Cohen-Boulakia, S., Thieffry, D., Paulevé, L.: The colomoto interactive notebook: Accessible and reproducible computational analyses for qualitative biological networks. Frontiers in Physiology **9**, 680 (2018). doi:10.3389/fphys.2018.00680
38. Naldi, A., Monteiro, P.T., Müssel, C., for Logical Models, C., Tools, Kestler, H.A., Thieffry, D., Xenarios, I., Saez-Rodriguez, J., Helikar, T., Chaouiya, C.: Cooperative development of logical modelling standards and tools with colomoto. Bioinformatics **31**(7), 1154–1159 (2015)
39. Klarner, H., Streck, A., Siebert, H.: Pyboolnet: a python package for the generation, analysis and visualization of boolean networks. Bioinformatics **33**(5), 770–772 (2017)
40. Chaouiya, C., Naldi, A., Thieffry, D.: Logical modelling of gene regulatory networks with GINsim **804**, 463–479 (2012)
41. Di Cara, A., Garg, A., De Micheli, G., Xenarios, I., Mendoza, L.: Dynamic simulation of regulatory networks using squad. BMC bioinformatics **8**(1), 462 (2007)
42. Malik-Sheriff, R.S., Glont, M., Nguyen, T.V.N., Tiwari, K., Roberts, M.G., Xavier, A., Vu, M.T., Men, J., Maire, M., Kananathan, S., Fairbanks, E.L., Meyer, J.P., Arankalle, C., Varusai, T.M., Knight-Schrijver, V., Li, L., Dueñas-Roca, C., Dass, G., Keating, S.M., Park, Y.M., Buso, N., Rodriguez, N., Hucka, M., Hermjakob, H.: BioModels — 15 years of sharing computational models in life science. Nucleic Acids Research **48**(D1), 407–415 (2020). doi:10.1093/nar/gkz1055. gkz1055. https://academic.oup.com/nar/article-pdf/48/D1/D407/31698010/gkz1055.pdff
43. Fauré, A., Vreede, B., Sucena, E., Chaouiya, C.: A discrete model of drosophila eggshell patterning reveals cell-autonomous and juxtacrine effects. PLoS Comput Biol **10**, 1003527 (2014). doi:10.1371/journal.pcbi.1003527
44. Delaplace, F., Ivanov, S.: Bisimilar booleanization of multivalued networks. BioSystems, 104205 (2020)
45. Grieco, L., Calzone, L., Bernard-Pierrot, I., Radvanyi, F., Kahn-Perles, B., Thieffry, D.: Integrative modelling of the influence of mapk network on cancer cell fate decision. PLoS Comput Biol **9**(10), 1003286 (2013)
46. Coolen, M., Thieffry, D., Drivenes, Ø., Becker, T.S., Bally-Cuif, L.: mir-9 controls the timing of neurogenesis through the direct inhibition of antagonistic factors. Developmental cell **22**(5), 1052–1064 (2012)
47. Sipser, M.: Introduction to the Theory of Computation, 3rd edn. Course Technology, Boston, MA (2013)
48. Wittmann, D.M., Krumsiek, J., Saez-Rodriguez, J., Lauffenburger, D.A., Klamt, S., Theis, F.J.: Transforming boolean models to continuous models: methodology and application to T-cell receptor signaling. BMC Systems Biology **3**(1), 98 (2009). doi:10.1186/1752-0509-3-98
49. Müssel, C., Hopfensitz, M., Kestler, H.A.: Boolnet—an r package for generation, reconstruction and analysis of boolean networks. Bioinformatics **26**(10), 1378–1380 (2010)
50. Mbodj, A., Junion, G., Brun, C., Furlong, E.E., Thieffry, D.: Logical modelling of drosophila signalling pathways. Molecular BioSystems **9**(9), 2248–2258 (2013)
51. Martinez-Sanchez, M.E., Hiriart, M., Alvarez-Buylla, E.R.: The cd4+ t cell regulatory network mediates inflammatory responses during acute hyperinsulinemia: a simulation study. BMC systems biology **11**(1), 1–12 (2017)
52. Mbodj, A., Gustafson, E.H., Ciglar, L., Junion, G., Gonzalez, A., Girardot, C., Perrin, L., Furlong, E.E., Thieffry, D.: Qualitative dynamical modelling can formally explain mesoderm specification and predict novel developmental phenotypes. PLoS computational biology **12**(9), 1005073 (2016)
53. Martinez-Sanchez, M.E., Mendoza, L., Villarreal, C., Alvarez-Buylla, E.R.: A minimal regulatory network of extrinsic and intrinsic factors recovers observed patterns of cd4+ t cell differentiation and plasticity. PLoS computational biology **11**(6), 1004324 (2015)
54. Fauré, A., Vreede, B.M., Sucena, É., Chaouiya, C.: A discrete model of drosophila eggshell patterning reveals cell-autonomous and juxtacrine effects. PLoS Comput Biol **10**(3), 1003527 (2014)
55. Sánchez, L., Chaouiya, C.: Primary sex determination of placental mammals: a modelling study uncovers dynamical developmental constraints in the formation of sertoli and granulosa cells. BMC systems biology **10**(1), 1–11 (2016)
56. Mombach, J.C., Bugs, C.A., Chaouiya, C.: Modelling the onset of senescence at the g1/s cell cycle checkpoint. BMC genomics **15**(S7), 7 (2014)
57. Corral-Jara, K.F., Chauvin, C., Abou-Jaoudé, W., Grandclaudon, M., Naldi, A., Soumelis, V., Thieffry, D.: Interplay between smad2 and stat5a is a critical determinant of il-17a/il-17f differential expression. Molecular Biomedicine **2**(1), 1–16 (2021)
58. Abou-Jaoudé, W., Monteiro, P.T., Naldi, A., Grandclaudon, M., Soumelis, V., Chaouiya, C., Thieffry, D.: Model checking to assess t-helper cell plasticity. Frontiers in bioengineering and biotechnology **2**, 86 (2015)
59. Kondratova, M., Barillot, E., Zinovyev, A., Calzone, L.: Modelling of immune checkpoint network explains synergistic effects of combined immune checkpoint inhibitor therapy and the impact of cytokines in patient

response. Cancers **12**(12), 3600 (2020)
60. Vaga, S., Bernardo-Faura, M., Cokelaer, T., Maiolica, A., Barnes, C.A., Gillet, L.C., Hegemann, B., van Drogen, F., Sharifian, H., Klipp, E., *et al.*: Phosphoproteomic analyses reveal novel cross-modulation mechanisms between two signaling pathways in yeast. Molecular systems biology **10**(12), 767 (2014)
61. Naldi, A., Carneiro, J., Chaouiya, C., Thieffry, D.: Diversity and plasticity of th cell types predicted from regulatory network modelling. PLoS Comput Biol **6**(9), 1000912 (2010)
62. Nuñez-Reza, K.J., Naldi, A., Sánchez-Jiménez, A., Leon-Apodaca, A.V., Santana, M.A., Thomas-Chollier, M., Thieffry, D., Medina-Rivera, A.: Logical modelling of in vitro differentiation of human monocytes into dendritic cells unravels novel transcriptional regulatory interactions. Interface focus **11**(4), 20200061 (2021)
63. Terfve, C., Cokelaer, T., Henriques, D., MacNamara, A., Goncalves, E., Morris, M.K., Iersel, M.v., Lauffenburger, D.A., Saez-Rodriguez, J.: Cellnoptr: a flexible toolkit to train protein signaling networks to data using multiple logic formalisms. BMC systems biology **6**(1), 1–14 (2012)
64. Floc'Hlay, S., Molina, M.D., Hernandez, C., Haillot, E., Thomas-Chollier, M., Lepage, T., Thieffry, D.: Deciphering and modelling the tgf-β signalling interplays specifying the dorsal-ventral axis of the sea urchin embryo. Development **148**(2), 189944 (2021)
65. Hernandez, C., Thomas-Chollier, M., Naldi, A., Thieffry, D.: Computational verification of large logical models-application to the prediction of t cell response to checkpoint inhibitors. bioRxiv (2020)
66. Fauré, A., Naldi, A., Lopez, F., Chaouiya, C., Ciliberto, A., Thieffry, D.: Modular logical modelling of the budding yeast cell cycle. Molecular BioSystems **5**(12), 1787–1796 (2009)
67. Calzone, L., Tournier, L., Fourquet, S., Thieffry, D., Zhivotovsky, B., Barillot, E., Zinovyev, A.: Mathematical modelling of cell-fate decision in response to death receptor engagement. PLoS Comput Biol **6**(3), 1000702 (2010)
68. Niarakis, A., Bounab, Y., Grieco, L., Roncagalli, R., Hesse, A.-M., Garin, J., Malissen, B., Daëron, M., Thieffry, D.: Computational modeling of the main signaling pathways involved in mast cell activation. Fc Receptors, 69–93 (2014)
69. Sahin, Ö., Fröhlich, H., Löbke, C., Korf, U., Burmester, S., Majety, M., Mattern, J., Schupp, I., Chaouiya, C., Thieffry, D., *et al.*: Modeling erbb receptor-regulated g1/s transition to find novel targets for de novo trastuzumab resistance. BMC systems biology **3**(1), 1 (2009)
70. MacNamara, A., Terfve, C., Henriques, D., Bernabé, B.P., Saez-Rodriguez, J.: State–time spectrum of signal transduction logic models. Physical biology **9**(4), 045003 (2012)
71. Selvaggio, G., Canato, S., Pawar, A., Monteiro, P.T., Guerreiro, P.S., Brás, M.M., Janody, F., Chaouiya, C.: Hybrid epithelial–mesenchymal phenotypes are controlled by microenvironmental factors. Cancer Research **80**(11), 2407–2420 (2020)
72. Chaouiya, C., Bérenguier, D., Keating, S.M., Naldi, A., Van Iersel, M.P., Rodriguez, N., Dräger, A., Büchel, F., Cokelaer, T., Kowal, B., *et al.*: SBML qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools. BMC systems biology **7**(1), 1–15 (2013)
73. Remy, E., Rebouissou, S., Chaouiya, C., Zinovyev, A., Radvanyi, F., Calzone, L.: A modeling approach to explain mutually exclusive and co-occurring genetic alterations in bladder tumorigenesis. Cancer research **75**(19), 4042–4052 (2015)
74. González, A., Chaouiya, C., Thieffry, D.: Dynamical analysis of the regulatory network defining the dorsal–ventral boundary of the drosophila wing imaginal disc. Genetics **174**(3), 1625–1634 (2006)
75. Sánchez, L., Chaouiya, C.: Logical modelling uncovers developmental constraints for primary sex determination of chicken gonads. Journal of The Royal Society Interface **15**(142), 20180165 (2018)
76. Hamey, F.K., Nestorowa, S., Kinston, S.J., Kent, D.G., Wilson, N.K., Göttgens, B.: Reconstructing blood stem cell regulatory network models from single-cell molecular profiles. Proceedings of the National Academy of Sciences **114**(23), 5822–5829 (2017)
77. Béal, J., Pantolini, L., Noël, V., Barillot, E., Calzone, L.: Personalized logical models to investigate cancer response to braf treatments in melanomas and colorectal cancers. PLOS Computational Biology **17**(1), 1007900 (2021)
78. Simao, E., Remy, E., Thieffry, D., Chaouiya, C.: Qualitative modelling of regulated metabolic pathways: application to the tryptophan biosynthesis in e. coli. Bioinformatics **21**(suppl_2), 190–196 (2005)
79. Enciso, J., Mayani, H., Mendoza, L., Pelayo, R.: Modeling the pro-inflammatory tumor microenvironment in acute lymphoblastic leukemia predicts a breakdown of hematopoietic-mesenchymal communication networks. Frontiers in physiology **7**, 349 (2016)
80. Sánchez, L., Thieffry, D.: A logical analysis of the drosophila gap-gene system. Journal of theoretical Biology **211**(2), 115–141 (2001)
81. Fauré, A., Thieffry, D.: Logical modelling of cell cycle control in eukaryotes: a comparative study. Molecular BioSystems **5**(12), 1569–1581 (2009)
82. Mendoza, L.: A network model for the control of the differentiation process in th cells. Biosystems **84**(2), 101–114 (2006)
83. González, A., Chaouiya, C., Thieffry, D.: Logical modelling of the role of the hh pathway in the patterning of the drosophila wing disc. Bioinformatics **24**(16), 234–240 (2008)
84. Sánchez, L., Chaouiya, C., Thieffry, D.: Segmenting the fly embryo: logical analysis of the role of the segment polarity cross-regulatory module. International journal of developmental biology **52**(8), 1059–1075 (2002)
85. Montagud, A., Béal, J., Tobalina, L., Traynard, P., Subramanian, V., Szalai, B., Alföldi, R., Puskás, L., Valencia, A., Barillot, E., Saez-Rodriguez, J., Calzone, L.: Patient-specific boolean models of signaling networks guide personalized treatments. bioRxiv (2021). doi:10.1101/2021.07.28.454126. https://www.biorxiv.org/content/early/2021/07/29/2021.07.28.454126.full.pdf
86. Sánchez-Villanueva, J.A., Rodríguez-Jorge, O., Ramírez-Pliego, O., Rosas Salgado, G., Abou-Jaoudé, W., Hernandez, C., Naldi, A., Thieffry, D., Santana, M.A.: Contribution of ros and metabolic status to neonatal and adult cd8+ t cell activation. PloS one **14**(12), 0226388 (2019)
87. Verlingue, L., Dugourd, A., Stoll, G., Barillot, E., Calzone, L., Londoño-Vallejo, A.: A comprehensive approach

to the molecular determinants of lifespan using a boolean model of geroconversion. Aging cell **15**(6), 1018–1026 (2016)
88. Flobak, Å., Baudot, A., Remy, E., Thommesen, L., Thieffry, D., Kuiper, M., Lægreid, A.: Discovery of drug synergies in gastric cancer cells predicted by logical modeling. PLoS Comput Biol **11**(8), 1004426 (2015)
89. Zañudo, J.G., Steinway, S.N., Albert, R.: Discrete dynamic network modeling of oncogenic signaling: Mechanistic insights for personalized treatment of cancer. Current Opinion in Systems Biology **9**, 1–10 (2018)
90. Cohen, D.P., Martignetti, L., Robine, S., Barillot, E., Zinovyev, A., Calzone, L.: Mathematical modelling of molecular pathways enabling tumour cell invasion and migration. PLoS Comput Biol **11**(11), 1004571 (2015)
91. Cacace, E., Collombet, S., Thieffry, D.: Logical modeling of cell fate specification—application to t cell commitment **139**, 205–238 (2020)
92. Collombet, S., van Oevelen, C., Ortega, J.L.S., Abou-Jaoudé, W., Di Stefano, B., Thomas-Chollier, M., Graf, T., Thieffry, D.: Logical modeling of lymphoid and myeloid cell specification and transdifferentiation. Proceedings of the National Academy of Sciences **114**(23), 5792–5799 (2017)
93. Traynard, P., Fauré, A., Fages, F., Thieffry, D.: Logical model specification aided by model-checking techniques: application to the mammalian cell cycle regulation. Bioinformatics **32**(17), 772–780 (2016)
94. Abou-Jaoudé, W., Ouattara, D.A., Kaufman, M.: From structure to dynamics: frequency tuning in the p53–mdm2 network: I. logical approach. Journal of theoretical biology **258**(4), 561–577 (2009)

1 Technical Results

1.1 Preliminaries

First, we formalize a BN as follows.

Definition 1 (Boolean Network (BN)) *A BN is a pair (X, F) where $X = \{x_1, ..., x_n\}$ is a set of variables and $F = \{f_{x_1}, ..., f_{x_n}\}$ is a set of update functions, with $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$ being the update function of variable x_i.*

The *state* of a BN is an evaluation of the variables, denoted by a vector of values $\mathbf{s} = (s_{x_1}, \ldots, s_{x_n}) \in \mathbb{B}^n$. Given a *partition of synchronization* $\{K_1, \ldots, K_m\} = \mathcal{K}$ of the variables X, and two states for $\mathbf{s}, \mathbf{t} \in \mathbb{B}^n$, we have a transition $\mathbf{s} \to \mathbf{t}$ if there exists a block K in $\mathcal{K}$ such that

- $t_{x_i} = f_{x_i}(\mathbf{s})$ for all $x_i \in K$
- $t_{x_i} = s_{x_i}$ for all $x_i \notin K$

We next introduce the *state transition graph* of a BN with respect to a given partition of synchronization. This is a graph having all possible states as vertices, and all transition among states as edges.

Definition 2 (State transition graph (STG)) *Let $B = (X, F)$ be a BN and $\mathcal{K}$ a partition of X. The state transition graph of B w.r.t. the synchronization partition $\mathcal{K}$, denoted by $STG_{\mathcal{K}}(B)$, is a pair $(S, T_{\mathcal{K}})$, where $S = \mathbb{B}^n$ is the set vertices, while the set of transitions $T_{\mathcal{K}}$ is defined by*

$$T_{\mathcal{K}} = \{\boldsymbol{s} \to \boldsymbol{t} \mid \boldsymbol{t}_{|K} = F_{|K}(\boldsymbol{s}) \text{ and } \boldsymbol{t}_{|X \setminus K} = \boldsymbol{s}_{|X \setminus K} \text{ for some } \boldsymbol{s} \in S \text{ and } K \in \mathcal{K}\}.$$

Using common notation, $v_{|I}$ denotes the restriction of a vector v to the set of indices I. When $\mathcal{K}$ is clear from the context or does not have an impact on the statement, we shall drop the subscript $\mathcal{K}$.

We note that $(S, T_{\mathcal{K}})$ corresponds to the STG of a synchronous BN when $\mathcal{K} = \{X\} = \mathcal{K}_{sync}$ and to that of an asynchronous BN when $\mathcal{K} = \{\{x\} \mid x \in X\} = \mathcal{K}_{async}$. The case when $\mathcal{K}$ refines $\mathcal{K}_{sync}$ and is at the same time coarser than $\mathcal{K}_{async}$, instead, describes a middle ground where different sets of variables, the blocks of $\mathcal{K}$, update synchronously within their block, and asynchronously with respect to the other blocks. We call $\mathcal{K}$ *synchronization partition* because the updates of two variables are synchronized if and only if they belong to the same block of $\mathcal{K}$. Notably, this synchronization schema is supported, e.g., by popular BN analysis tools like GINsim [6] under the notion of *priority classes* as described in [7].

We shall use the notation $\mathbf{s} \to^+ \mathbf{t}$ for the transitive closure of the transition relation. With this, we can formally define the notion of attractors.

Definition 3 (Attractor) *Let $B = (X, F)$ be a BN with $STG(B) = (S, T)$. We say that a set of states $A \subseteq S$ is an* attractor *whenever*

1. $\forall \boldsymbol{s}, \boldsymbol{s}' \in A,\ \boldsymbol{s} \to^+ \boldsymbol{s}'$, *and*
2. $\forall \boldsymbol{s} \in A, \forall \boldsymbol{s}' \in S,\ \boldsymbol{s} \to^+ \boldsymbol{s}'$ *implies* $\boldsymbol{s}' \in A$.

Attractors are hence absorbing strongly connected components in the STG. An attractor A such that $|A| = 1$ is called a *steady state* (also named *point attractor*). We also denote with $|A|$ the *length* of attractor A.

1.2 Boolean Backward Equivalence

Let X be a set, and $\mathcal{H}$ a partition over it. Any partition obtained by breaking down the blocks of $\mathcal{H}$ into sub-blocks is said to be a refinement of $\mathcal{H}$. The notion of BBE, the algorithm for its computation, and the notion of BN reduced up to a BBE do not depend on the used synchronization partition $\mathcal{K}$. However, as we shall see, a BBE $\mathcal{H}$ guarantees the preservation of dynamics of a BN only if $\mathcal{H}$ refines $\mathcal{K}$. This can be guaranteed by using as initial partition $\mathcal{G}$ either $\mathcal{K}$, or any refinement of it.

We first introduce the notion of *constant* state on a partition $\mathcal{H}$.

Definition 4 (Constant State) *Let X be a set of variables, and $\mathcal{H}$ a partition of X. A state $\boldsymbol{s} \in \mathbb{B}^n$ is* constant *on $\mathcal{H}$ if and only if for all $H \in \mathcal{H}$ and $x_i, x_j \in H$ it holds that $s_{x_i} = s_{x_j}$.*

We now define the notion of BN reduced up to a BBE $\mathcal{H}$. Each variable in the reduced BN represents one block of $\mathcal{H}$. Informally, we pick one variable per block, select the update function of any variable in such block and replace all variables in it with the representative of the block the variable belongs to. Formally, we denote by $f[a/b]$ the term arising by replacing each occurrence of a by b in the function f.

Definition 5 (BN reduction) *The reduction of B up to $\mathcal{H}$, denoted by $B_{\mathcal{H}}$, is the BN $(X_{\mathcal{H}}, F_{\mathcal{H}})$ where $F_{\mathcal{H}} = \{f_{x_H} \mid H \in X_{\mathcal{H}}\}$ and, for any $H \in \mathcal{H}$ and some $x_k \in H$, one sets $f_{x_H} = f_{x_k}\left[x_i/x_{H'} \mid \forall H' \in X_{\mathcal{H}}, \forall x_i \in H'\right]$.*

Algorithm S1: Compute maximal BBE of (X, F) refining an initial partition $\mathcal{G}$

Result: maximal BBE $\mathcal{H}$ that refines an arbitrary partition $\mathcal{G}$
$\mathcal{H} \leftarrow \mathcal{G}$;
while *true* **do**
  if $\Phi^{\mathcal{H}}$ *is valid* **then**
    return $\mathcal{H}$;
  else
    $\mathbf{s} \leftarrow$ get a state that satisfies $\neg\Phi^{\mathcal{H}}$;
    $\mathcal{H}' \leftarrow \emptyset$;
    for $H \in \mathcal{H}$ **do**
      $H_0 = \{x_i \in H : f_{x_i}(\mathbf{s}) = 0\}$;
      $H_1 = \{x_i \in H : f_{x_i}(\mathbf{s}) = 1\}$;
      $\mathcal{H}' = \mathcal{H}' \cup \{H_1\} \cup \{H_0\}$;
    end
    $\mathcal{H} \leftarrow \mathcal{H}' \setminus \{\emptyset\}$;
  end
end

The partition refinement algorithm is shown in Algorithm S1. Its inputs are a BN and $\mathcal{G}$, an initial partition of its variables X. The output of the algorithm is the coarsest partition that is a BBE and that refines $\mathcal{G}$.
The number of iterations needed to reach a BBE depends on the state assignments that the SAT solver provides but is at most $|X| = n$ because a partition over X can be refined at most $|X|$ times. Each iteration requires to solve a SAT problem which is known to be NP-complete [47]. However, as discussed in the main text, our implementation can scale to the largest models present in popular BN repositories.
We first show that given an initial partition there exists a unique coarsest BBE.

Theorem 1 *Fix a BN (X, F) and a partition $\mathcal{G}$. There exists a unique maximal BBE $\mathcal{H}$ that refines $\mathcal{G}$.*

Proof of Theorem 1 Let $\mathcal{H}_1$, $\mathcal{H}_2$ be two BBE partitions that refine some other partition $\mathcal{G}$ that is not necessarily a BBE. Let R_1, R_2, R_3 be equivalence relations over X inducing $\mathcal{H}_1, \mathcal{H}_2$ and $\mathcal{G}$, respectively. We start by noting that $R = (R_1 \cup R_2)^* \subseteq R_3$, where the asterisk denotes the transitive closure. Hence, $\mathcal{X}_R$ is a refinement of $\mathcal{G}$, where $\mathcal{X}_R = X/R$. We next show that $\mathcal{X}_R$ is a BBE partition. To this end, fix some $\mathbf{s} \in \mathbb{B}^n$ that is constant on $\mathcal{X}_R$. Since $R_i \subseteq R$, this implies that $\mathbf{s}$ is constant on $\mathcal{X}_i$ which, in virtue of $\mathcal{H}_i$ being a BBE, implies that $F(\mathbf{s}) \in \mathbb{B}^n$ is constant on $\mathcal{H}_i$. This implies that $F(\mathbf{s}) \in \mathbb{B}^n$ is constant on $\mathcal{X}_R$, i.e., that $\mathcal{X}_R$ is indeed a BBE partition. The overall claim follows by noting that the finiteness of X implies that there are finitely many BBE partitions $\mathcal{H}_i$ that refine any given partition $\mathcal{G}$ of X. □

We now prove that Algorithm S1 provides indeed the maximal BBE that refines the initial one.

Theorem 2 *Algorithm S1 computes the maximal BBE partition refining $\mathcal{G}$.*

Proof of Theorem 2 Assume that $\mathcal{G}'$ denotes the coarsest BBE partition that refines some given partition $\mathcal{G}$. Set $\mathcal{H}_0 := \mathcal{G}$ and define for all $k \geq 0$

$$\mathcal{H}_{k+1} := \big(\{H_0 \mid H \in \mathcal{H}_k\} \cup \{H_1 \mid H \in \mathcal{H}_k\}\big) \setminus \{\emptyset\},$$

where H_0 and H_1 are as in Algorithm S1. Then, a proof by induction over $k \geq 1$ shows that (a) $\mathcal{G}'$ is a refinement of $\mathcal{H}_k$ and (b) $\mathcal{H}_k$ is a refinement of $\mathcal{H}_{k-1}$, for all $k \geq 1$. Since $\mathcal{G}'$ is a refinement of any $\mathcal{H}_k$, it holds that $\mathcal{G}' = \mathcal{H}_k$ if $\mathcal{H}_k$ is a BBE partition. Since X is finite, b) allows us to fix the smallest $k \geq 1$ such that $\mathcal{H}_k = \mathcal{H}_{k-1}$. This, in turn, implies that $\mathcal{H}_{k-1}$ is a BBE. □

1.3 Relating Dynamics of Original and Reduced BNs

We next relate the STGs of the original and the reduced BN.

Definition 6 *Fix a BN $B = (X, F)$, a BBE $\mathcal{H}$ of B, a synchronization partition $\mathcal{K}$, and $STG_{\mathcal{K}}(B) = (S, T_{\mathcal{K}})$ such that $\mathcal{K}$ is coarser than $\mathcal{H}$. With this, the STG of $B/\mathcal{H} = (X_{\mathcal{H}}, F_{\mathcal{H}})$ has synchronization partition $\mathcal{K}_{\mathcal{H}} = \big\{\{H_j \mid x_i \in K \text{ and } x_i \in H_j\} \mid K \in \mathcal{K}\big\}$ and states $m_{\mathcal{H}}(S_{|\mathcal{H}})$, where*

- *$S_{|\mathcal{H}}$ denotes all states of S constant on $\mathcal{H}$ and;*

- $m_{\mathcal{H}} : S_{|\mathcal{H}} \to S_{\mathcal{H}}$ *is given by* $m_{\mathcal{H}}(\boldsymbol{s}) = (v_{H_1}, \ldots, v_{H_{|\mathcal{H}|}})$ *and extends to sets via elementwise application, while* $v_{H_j} := s_{x_i}$ *for previously chosen representative* $x_i \in H_j$.

The following lemma ensures that all attractors of $STG_{\mathcal{K}}(B)$ containing states constant on $\mathcal{H}$ are preserved by $STG_{\mathcal{K}_{\mathcal{H}}}(B/\mathcal{H})$.

Lemma 3 (Constant attractors) *Fix a BN* $B = (X, F)$*, a BBE* $\mathcal{H}$ *of* B *and* $STG_{\mathcal{K}}(B) = (S, T_{\mathcal{K}})$ *such that* $\mathcal{K}$ *is coarser than* $\mathcal{H}$*. Let us further assume that* A *is an attractor of* $STG_{\mathcal{K}}(B)$*. With this, if* $A \cap S_{|\mathcal{H}} \neq \emptyset$*, then* $A \subseteq S_{|\mathcal{H}}$.

Proof of Lemma 3 By assumption, we can pick a state $\mathbf{s} \in A$ that is constant on $\mathcal{H}$. The fact that $\mathcal{H}$ is a BBE refining $\mathcal{K}$ ensures that any state $\mathbf{t}$ with $\mathbf{s} \to^+ \mathbf{t}$ is also constant on $\mathcal{H}$. Actually, it is trivial to show that $A = \{\mathbf{t} \mid \mathbf{s} \to^+ \mathbf{t}\}$, thus implying that $A \subseteq S_{|\mathcal{H}}$. □

The next proposition ensures that BBE does not generate spurious trajectories or attractors in the reduced system. In particular we show that the STG of the reduced BN is a subgraph (modulo state renaming) of the STG of the original BN.

Proposition 4 (Reduction isomorphism) *Fix a BN* $B = (X, F)$*, a BBE* $\mathcal{H}$ *of* B *and* $STG_{\mathcal{K}}(B) = (S, T_{\mathcal{K}})$ *such that* $\mathcal{K}$ *is coarser than* $\mathcal{H}$*. It can be shown that* $STG_{\mathcal{K}_{\mathcal{H}}}(B/\mathcal{H})$ *is described by* $\big(m_{\mathcal{H}}(S_{|\mathcal{H}}), m_{\mathcal{H}}(T_{\mathcal{K}} \cap (S_{|\mathcal{H}} \times S_{|\mathcal{H}}))\big)$*. Furthermore*

1. *For all states* $\boldsymbol{s} \in S_{|\mathcal{H}}$ *it holds* $F_{\mathcal{H}}(m_{\mathcal{H}}(\boldsymbol{s})) = m_{\mathcal{H}}(F(\boldsymbol{s}))$.
2. *For all states* $\boldsymbol{s} \in S_{\mathcal{H}}$ *it holds* $F(m_{\mathcal{H}}^{-1}(\boldsymbol{s})) = m_{\mathcal{H}}^{-1}(F_{\mathcal{H}}(\boldsymbol{s}))$.

Proof of Proposition 4 Follows readily from the definition of a BBE, $STG_{\mathcal{K}_{\mathcal{H}}}(B/\mathcal{H})$, and $m_{\mathcal{H}}$. □

Instead, the following example shows that it is necessary for the initial partition to be a refinement of the synchronization partition of the model.

Example 1 *Let us consider the 3-variables example from Fig. 1. Let us assume that the model is equipped with the synchronization partition* $\mathcal{K} = \{\{x_1\}, \{x_2, x_3\}\}$*. This means, e.g., that from state* 000 *we can go either in state* 100 *by updating* x_1*, or in state* 010*. From both states, we can go to state* 110*. If we apply BBE using the initial partition* $\mathcal{H} = \{\{x_1, x_2, x_3\}\}$ *that does not refine* $\mathcal{K}$*, we get the same reduced model as in Fig. 1. In such reduced model, we find the reduced variable* $x_{1,2}$ *representing variables* x_1 *and* x_2 *which, however, shall not be updated synchronously according to* $\mathcal{K}$*. Therefore, it is not possible to define the synchronization partition* $\mathcal{K}_{\mathcal{H}}$ *as given in Definition 6. Note furthermore that if we opt for a synchronization partition enabling the synchronous update of* $x_{1,2}$ *and* x_3*, we get the STG from the top-right of Fig. 1. Here, our reduction isomorphism result does not hold, because the reduced STG cannot express the above-discussed 2-steps path from 000 to 110. In fact, the corresponding path from 00 to 10 is done in only 1 transition.*

We can now state the main result of our approach, namely that the BBE reduction of a BN for a BBE $\mathcal{H}$ exactly preserves all attractors that are constant on $\mathcal{H}$ up to renaming with $m_{\mathcal{H}}$.

Theorem 5 (Constant attractor preservation) *Fix a BN* $B = (X, F)$*, a BBE* $\mathcal{H}$ *of* B *and* $STG_{\mathcal{K}}(B) = (S, T_{\mathcal{K}})$ *such that* $\mathcal{K}$ *is coarser than* $\mathcal{H}$*. Let us further assume that* A *is an attractor of* $STG_{\mathcal{K}}(B)$*. With this, if* $A \cap S_{|\mathcal{H}} \neq \emptyset$*, then* $m_{\mathcal{H}}(A)$ *is an attractor of* $STG_{\mathcal{K}}(B)$*. Furthermore, given a state* $\boldsymbol{s} \in S_{|\mathcal{H}}$ *and an attractor* A *such that* $A \cap S_{|\mathcal{H}} \neq \emptyset$*, we have that* A *is reachable from* $\boldsymbol{s}$ *if and only if* $m_{\mathcal{H}}(A)$ *is reachable from* $m_{\mathcal{H}}(\boldsymbol{s})$.

Proof of Theorem 5 The theorem readily follows from Lemma 3 and Proposition 4. □

2 Comparison with encoding-based reductions

In this section we discuss how BN reduction techniques mediated by a translation into another formalism may miss certain reductions. In particular, we present a comparison with the approach from [19] based on ordinary differential equations (ODEs). Further details on such comparison can be found in [34]. The considered ODE-based approach first applies a so-called *odification* technique to encode a BN into an ODE system [48]; then it applies backward equivalence to ODEs, which is the ODE counterpart of BBE.

We consider the *TCR-TLR* model from [30], part of which adopted in the *Method* section. The equations for two of the variables in the model, *MyD88* and *IRAK4*, are given by:

$$x_{MyD88}(t+1) = x_{TLR5}(t)$$
$$x_{IRAK4}(t+1) = (\neg x_{MyD88}(t) \wedge x_{TICAM1}(t)) \vee (x_{MyD88}(t))$$

Using maximal reduction, BBE reveals that these two variables are equivalent because also *TICAM1* and *TLR5* are so. The corresponding ODEs after odification are instead given by:

$$x'_{MyD88} = x_{TLR5} - x_{MyD88}$$
$$x'_{IRAK4} = x_{MyD88} + x_{TICAM1} - x_{MyD88} \cdot x_{TICAM1} - x_{IRAK4}$$

where x' denotes the derivative of variable x with respect to time. The ODE variables for *TLR5*, *MyD88*, and *TICAM1* can shown to be still backward equivalent. However, differently from BBE, *IRAK4* is not anymore ODE backward equivalent to the others. Indeed, since ODEs allow a continuous range of values in the interval $[0; 1]$, the property that the solution of variables must be equal at all time points must be valid for all possible such values. However, if all variables have value 0.5, then we get derivative with value 0 for *MyD88* and value 0.25 for *IRAK4*, which indeed makes them not ODE backward equivalent.

3 Application of BBE to a BN with partially asynchronous schema

We present an application of BBE to a BN with partially asynchronous schema. According to Section 1, here we use a partition of synchronisation $\mathcal{K}$ separating variables in blocks. At each time point, one block $K \in \mathcal{K}$ is non-deterministically selected, and all and only the variables in K are updated syncrhonously. As mentioned in previous sections, this type of synchronization schema is supported, e.g., by popular BN analysis tools like GINsim [6] under the notion of *"priority classes"* [7]. We focus on the BN of [46] which is displayed in the left part of Fig. S1. The BN models neurogenesis: the process by which nervous system cells, the neurons, are produced by neural stem cells.

$$
\begin{aligned}
x_{Her6}(t+1) &= \neg x_{miR9}(t) \wedge \neg x_N(t)\\
x_{HuC}(t+1) &= \neg x_{miR9}(t) \wedge \neg x_P(t)\\
x_N(t+1) &= x_{HuC}(t)\\
x_P(t+1) &= x_{Her6}(t) \vee x_{Zic5}(t)\\
x_{Zic5}(t+1) &= \neg x_{miR9}(t) \wedge \neg x_N(t)\\
x_{miR9}(t+1) &= \neg x_{Her6}(t) \wedge \neg x_N(t)
\end{aligned}
\qquad
\begin{aligned}
x_{\{Her6,Zic5\}}(t+1) &= \neg x_{\{miR9\}}(t) \wedge \neg x_{\{N\}}(t)\\
x_{\{HuC\}}(t+1) &= \neg x_{\{miR9\}}(t) \wedge \neg x_{\{P\}}(t)\\
x_{\{N\}}(t+1) &= x_{\{HuC\}}(t)\\
x_{\{P\}}(t+1) &= x_{\{Her6,Zic5\}}(t)\\
x_{\{miR9\}}(t+1) &= \neg x_{\{Her6,Zic5\}}(t) \wedge \neg x_{\{N\}}(t)
\end{aligned}
$$

Figure S1: (Left) The variables and update functions. (Right) The reduced BN obtained after collapsing the variables x_{Her6}, x_{Zic5} into one single variable component $x_{\{Her6,Zic5\}}$.

Hypothesis. The authors consider a fully synchronous schema wherein all variables are updated at the same time. However, the set $\{x_{Her6}, x_{Zic5}, x_P\}$ seems to update synchronously in vivo, while x_{miR9}, x_{HuC} and x_N update asynchronously both with the set $\{x_{Her6}, x_{Zic5}, x_P\}$, and with each other [46],

Configuration. To this end, we create a corresponding partition of synchronization as follows:

$$\mathcal{K} = \{\{x_{Her6}, x_{Zic5}, x_P\}, \{x_{miR9}\}, \{x_{HuC}\}, \{x_N\}\},$$

The STG according to this partition of synchronization is given in Fig. S2. This STG has been obtained using the GINsim tool.
We reduce this model using BBE. In order to be coherent with our theory, we set $\mathcal{K}$ as the initial partition for our reduction algorithm Algorithm S1. This enables us to use the results on preservation of dynamics from Section 1.

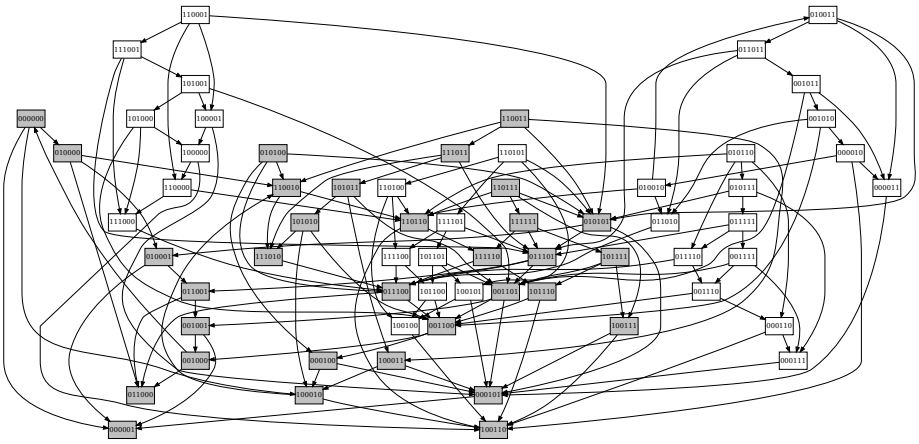


Figure S2: The STG of the BN of Fig. S1 (left) according to the partition of synchronization $\mathcal{K} = \{\{x_{Her6}, x_{Zic5}, x_P\}, \{x_{miR9}\}, \{x_{HuC}\}, \{x_N\}\}$.

Results. The resulting BBE is:

$$\{\{x_{Her6}, x_{Zic5}\}, \{x_P\}, \{x_{miR9}\}, \{x_{HuC}\}, \{x_N\}\}$$

Please note that, in this specific example, the update functions of the two related variables (x_{Her6}, and x_{Zic5}) have the same update function. However, we have seen in other examples here that BBE might relate also variables with apparently unrelated update functions (e.g., the case of x_{MyD88} and x_{IRAK4} in Section 2).

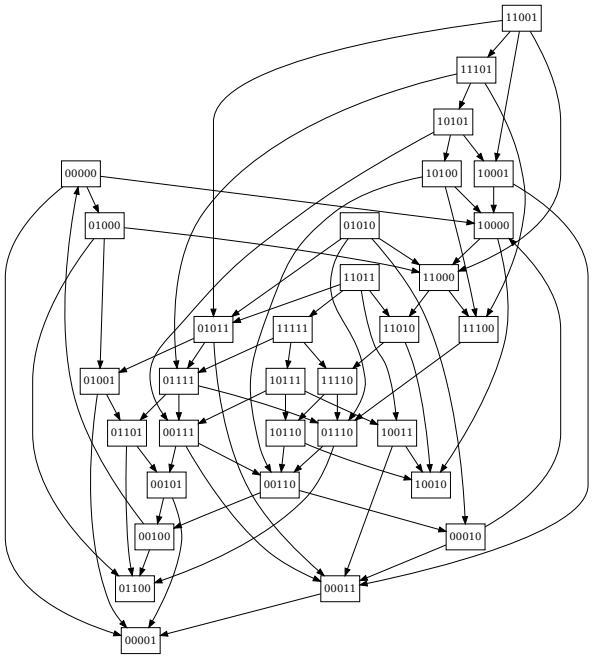
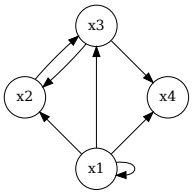


Figure S3: The STG of the BBE-reduced BN. This STG corresponds to the original one (Fig. S2) after dropping all states non constant on the used BBE. Note that the first and the fifth digit of each preserved state are collapsed into one single digit (these two digits are always equal in the preserved states).

The grey states of Fig. S2 correspond to the constant states of this BBE partition and, consequently, these are the preserved states after reduction. We obtain the reduced BN by collapsing $\{x_{Her6}, x_{Zic5}\}$ into a single variable component represented by the variable $x_{\{Her6,Zic5\}}$. The reduced BN is displayed in the right part of Fig. S1.

Interpretation and Discussion. The STG of the reduced BN is displayed in Fig. S3. The reduction isomorphism (Proposition 4) guarantees that the constant states are preserved with all the transitions between them, and that there are no transitions from the constant states to the non-constant states. According to Theorem 5, constant attractors are preserved (i.e., attractors containing at least one state constant on the BBE). In this case, the original BN has 3 steady states (the states 011000, 000001 and 100110) and all of them are preserved in the reduced BN.



$$
\begin{aligned}
x_1(t+1) &= x_1(t) \\
x_2(t+1) &= x_1(t) \vee x_3(t) \\
x_3(t+1) &= x_1(t) \wedge x_2(t) \\
x_4(t+1) &= \neg x_1(t) \wedge x_3(t)
\end{aligned}
$$

Figure S4: (Left) A randomly generated graph of influences for $n = 4, k = 2$. (Right) A BN with randomly generated update functions coherent with (Left).

4 Application of BBE to randomly generated Boolean Networks

In this section, we apply BBE to randomly generated BNs. These have been constructed by using an n-k model [49] as described in Kauffman's seminal work on BNs [1]. In partiuclar, n refers to the number of variables in the generated BNs, while k to the number of incoming influences of each variable. The process is described in Fig. S4: we first obtain a directed graph on n nodes. For each node, the number of incoming edges is drawn randomly from a Poisson distribution with mean k, choosing the source nodes randomly (see left part of Fig. S4). On average, the nodes of such randomly generated BNs will have k incoming edges. The nodes are then transformed in BN variables by using a procedure specified in [49] to randomly generate update functions coherent with the previously generated graph of influences (right part of Fig. S4). The procedure is implemented in the R package BoolNet [49]. In what comes later we will study BNs generated by varying both n (size of the BN) and k (*density* of the BN). For the additional parameters of the package not mentioned here, we use default values from [49].

Purpose. Our purpose is to investigate the scalability of BBE to randomly generated BNs as the number of variables increases, and estimate the expected loss of attractors. We consider two different values for k: 2, and 1, studying BBE at the varying of the density of influences in the BNs. [4]

Configuration. For $k = 2$, we generate 100 BNs for $n = 50$, $n = 100$, and $n = 200$ variables, resulting in 300 BNs overall. As done in the main text, we reduce these BNs using maximal and IS initial partitions, and compute the reduction ratios (paragraph *"Results on Reduction Magnitude"*). We also compute the number of attractors in the original and reduced BNs (paragraph *"Results related to attractor preservation."*). We then repeat the same analysis for $k = 1$, considering 300 more BNs. Overall, we consider 600 randomly generated BNs.

Results on Reduction Magnitude. As a reminder, the reduction ratio is defined as the fraction of the number of variables in the reduced BN, over the number of variables in the original BN. We display the reduction ratio for these 300 BNs for varying size and $k = 2$ in Fig S5. Both scenarios (IS and maximal) lead to the reduction of 299 out of the 300 BNs considered. Only one BN with $n = 50$ was not reduced (for any of the two initial partitions). When the red dot and the blue cross coincide, the IS and the maximal reduction have the same reduction ratio. Fig. S6 displays the same analysis for 300 networks of $k = 1$. In Table S1, we present the average reduction ratios of the BNs obtained for the different values of n and k. We can see that models generated for $k = 1$ allow for stronger reductions. We interpret this as follows: the more *sparse* is a BN (i.e., the less influences there are among the variables), the more effective becomes BBE.

	IS	*Maximal*	*IS*	*Maximal*
	k=2		k=1	
n=50	*0.878*	*0.875*	*0.556*	*0.542*
n=100	*0.856*	*0.852*	*0.517*	*0.502*
n=200	*0.837*	*0.833*	*0.464*	*0.450*

Table S1: Mean IS and maximal reduction ratio of the 600 randomly generated BNs.

In Table S2 (left) we provide the average reduction time, for IS and maximal initial partitions, of the 600 randomly generated BNs. We observe that the average reduction time seems to increase linearly with the size of the considered BN. In Table S2 (right) we display the maximum reduction time; in the worst case scenario the BBE-reduction was performed in about 3 seconds. The IS and maximal reductions seem to take about the same time.

Results related to attractor preservation. For $k = 2$, and $n = 100$ and 200, the tool BNS failed several times due to time-out (we imposed an arbitrary time-out of 30 minutes). Therefore, we focus only on the BNs with

[4] We chose $k = 2$ as maximum value because it was the largest value used in a similar study for a different analysis technique in [12] (section 6.2). In [12], further used values for k were 1.9, 1.889, and 1.875. Here we preferred to use $k = 1$ because we are interested in studying the effect on BBE of higher changes in the density of the interaction graph.

Average	*IS*	*Maximal*	*IS*	*Maximal*
	k=2		k=1	
n=50	*0.591*	*0.614*	*0.442*	*0.477*
n=100	*1.402*	*1.428*	*0.989*	*1.034*
n=200	*2.433*	*2.489*	*1.869*	*1.894*

Maximum	*IS*	*Maximal*	*IS*	*Maximal*
	k=2		k=1	
n=50	*0.870*	*0.943*	*0.923*	*0.901*
n=100	*1.888*	*1.907*	*1.437*	*1.478*
n=200	*3.285*	*3, 363*	*3.224*	*2.571*

Table S2: Mean (Left) and maximum (Right) reduction time of the 600 randomly generated BNs for IS and maximal initial partitions.

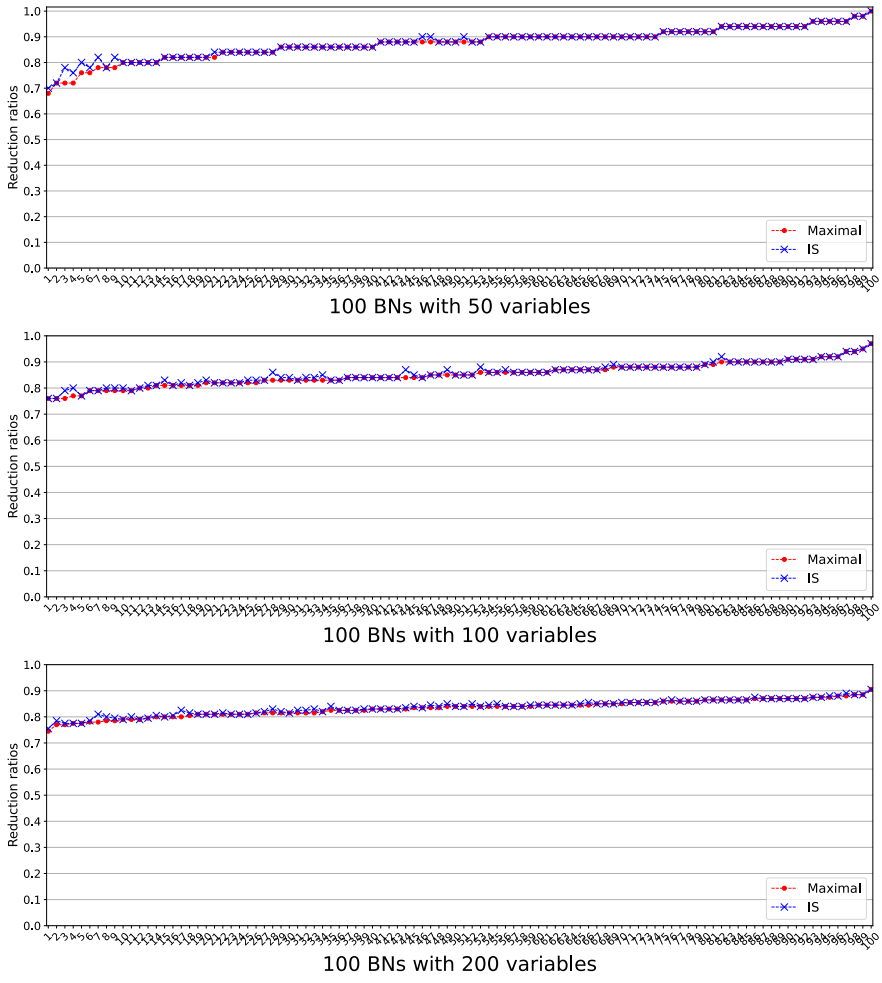


Figure S5: Reduction ratios for *300* randomly generated BNs using $k = 2$.

$n = 50$ for which we experienced only two time-outs (models 44 and 46 for which we report 10^{-1} as number of obtained attractors to stress that attractor generation failed). We consider only the 99 BNs that admitted BBE reduction. We display in Fig. S7 (top) the number of attractors in the original, the IS, and the maximal reduced BNs. In most cases, BBE preserves all attractors; the cases wherein attractors are lost are these where the orange line is above the other two lines (see, e.g., the BNs 7 and 55). The IS and maximal reduction scenario seemed to preserve the same number of attractors; these are cases wherein the red dot and the blue cross coincide. However, this case is not general as, for instance, BN 15 wherein the IS reduction preserves more attractors than the maximal one. The bottom part of Fig. S7 displays the corresponding information in the case of real BNs. Given the better reduction ratio obtained for the real models, here we tend to preserve a lower percentage of attractors. In $k = 1$, attractors generation succeeded always within the specified time limit of 30 minutes. The results are displayed in Fig. S8. Surprisingly, the much better reduction ratio than case $k = 2$ does not lead to lower preservation of attractors. Attractors are often fully preserved.

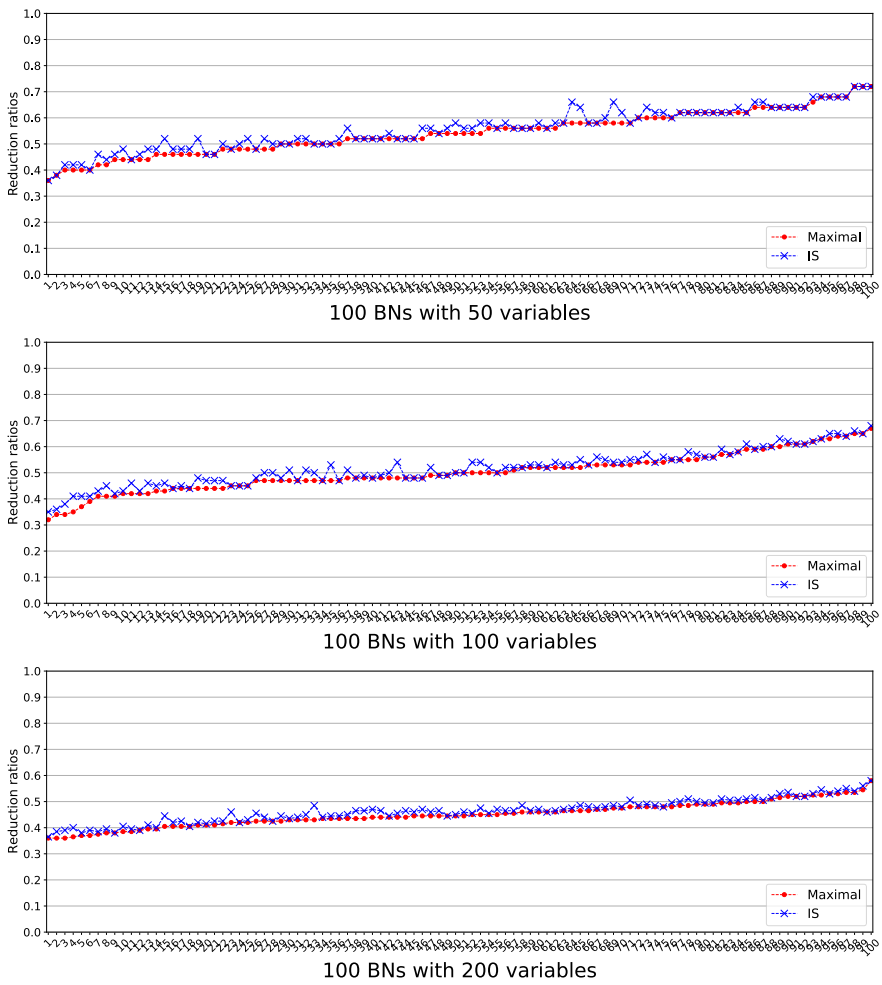


Figure S6: Reduction ratios for *300* randomly generated BNs using $k = 1$.

Interpretation. From Table S1, we see that BBE scales well with the size of BNs: while the number of variables increases, the reduction ratio decreases. Indeed, for $k = 2$ the average IS reduction ratio goes from 0.88 for $n = 50$, to 0.83 for $n = 200$. The same behavior is observed in the case of maximal reduction, and also in both cases for $k = 1$. The same table shows that the average reduction ratio is better when $k = 1$, meaning that BBE performs better when the density of the interaction graph of a BN (see Fig. S4, left) is low.
In the case of randomly generated BNs, attractors are often fully preserved. This is in contrast with the realistic BNs from the repositories. For $k = 2$ and $n = 100$ and $n = 200$, the BNS tool fails to compute the attractors within the 30-minutes time-out arbitrarily chosen by us. Here, several variables might have a high number of incoming influences. This leads to complex update Boolean functions that the BNS tool fails to manage. Instead, BBE terminated correctly on all randomly generated BNs in less than 3.5 seconds in all cases.
In Figs. S7, and S8 we have seen that for randomly generated BNs, BBE tends to preserve more attractors than for realistic BNs from the repositories. This might be reasonable for the cases $k = 2$, for which we have higher reduction ratios (we reduce less) than for the realistic BNs. Instead, this is somehow surprising for $k = 1$ where we reduce more. Given the persistent result obtained for $k = 2$ and $k = 1$ (and given that, as discussed, we considered a larger interval for k than in [12]), we suspect that this depends on oher parameters of the generation process not considered in our study. For example, a possible interpretation is that by looking at Figs. S7 and S8 we can see that the BNs from the repositories can generate more attractors (up to 10^5) than the randomly generated BNs (slightly above 10^2). However, we believe that a deeper study on the used generation process from the R package BoolNet is out of the scope of this paper.

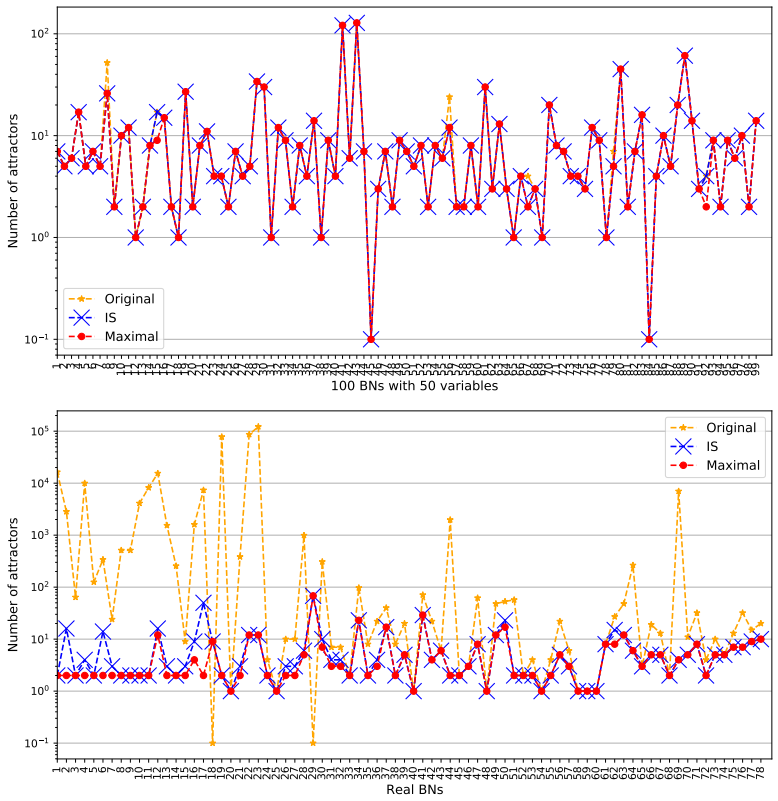


Figure S7: (Top) number of preserved attractors for 100 networks with 50 variables and $k = 2$. (Bottom) number of preserved attractors for the realistic BNs of Table S5 from the two online repositories. In bottom, for models 17 and 28 we provide "0 attractors" in the *Original* case to stress that it was not possible to compute the attractors for the original models 17 and 28 (while it was possible for their reductions). The number of attractors are given in log scale.

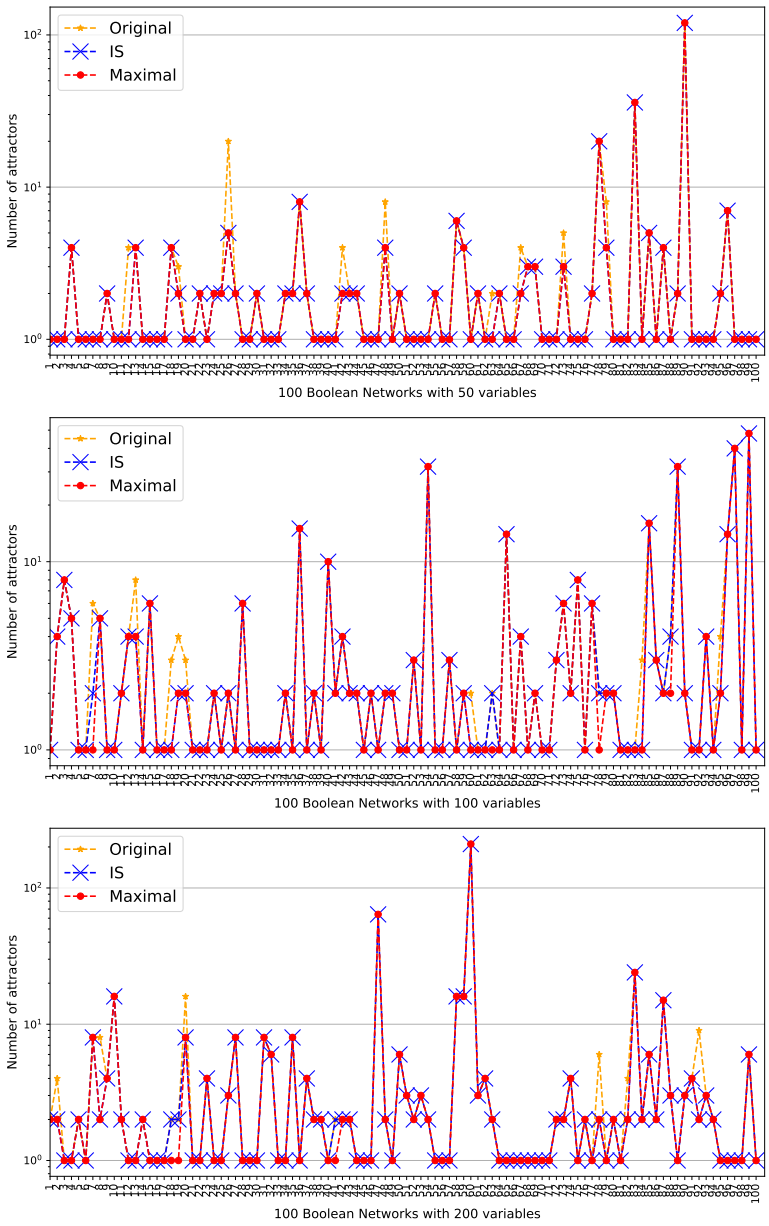


Figure S8: Number of preserved attractors for 300 randomly generated BNs of different sizes. In the generation process we set $k = 1$.

5 Tables

Model ID	URL	Size			Reduction Ratio	
		N	N_i	N_m	R_i	R_m
1 [50]	ginsim.org/node/97	26	7	4	26.92	15.38
2 [51]	www.ebi.ac.uk/biomodels/MODEL1606020000	19	11	3	57.89	15.79
3 [50]	ginsim.org/node/126	24	11	4	45.83	16.67
4 [52]	ginsim.org/model/drosophila_mesoderm	57	34	11	59.65	19.3
5 [53]	www.ebi.ac.uk/biomodels/BIOMD0000000592	10	2	2	20.0	20.0
6 [53]	www.ebi.ac.uk/biomodels/BIOMD0000000593	18	13	4	72.22	22.22
7 [54]	ginsim.org/model/eggshell_patterning	8	4	2	50.0	25.0
8 [50]	ginsim.org/node/115	16	8	5	50.0	31.25
9 [50]	ginsim.org/node/102	23	13	8	56.52	34.78
10 [50]	ginsim.org/node/109	19	8	7	42.11	36.84
11 [50]	ginsim.org/node/144	24	11	9	45.83	37.5
12 [55]	ginsim.org/model/sex_determination_mammals	37	16	14	43.24	37.84
13 [50]	ginsim.org/node/96	34	23	15	67.65	44.12
14 [50]	ginsim.org/node/160	18	11	8	61.11	44.44
15 [56]	ginsim.org/node/183	30	25	14	83.33	46.67
16 [54]	ginsim.org/model/eggshell_patterning	24	15	12	62.5	50.0
17 [57]	www.ebi.ac.uk/biomodels/MODEL2101150001	92	72	46	78.26	50.0
18 [58]	ginsim.org/node/185	103	54	52	52.43	50.49
19 [59]	ginsim.org/model/immune-checkpoints	66	43	35	65.15	53.03
20 [60]	www.ebi.ac.uk/biomodels/MODEL1506260002	45	24	24	53.33	53.33
21 [50]	ginsim.org/node/89	18	11	10	61.11	55.56
22 [61]	ginsim.org/node/79	36	23	21	63.89	58.33
23 [61]	ginsim.org/node/79	71	44	42	61.97	59.15
24 [62]	ginsim.org/model/monocytes-to-dc	96	81	57	84.38	59.38
25 [63]	www.ebi.ac.uk/biomodels/MODEL1506260000	82	49	49	59.76	59.76
26 [3]	www.ebi.ac.uk/biomodels/MODEL1504170000	10	7	6	70.0	60.0
27 [3]	www.ebi.ac.uk/biomodels/MODEL1504170003	10	7	6	70.0	60.0
28 [64]	ginsim.org/node/236	41	27	25	65.85	60.98
29 [65]	ginsim.org/model/tcell-checkpoint-inhibitors-tcla4-pd1	218	140	136	64.22	62.39
30 [55]	ginsim.org/model/sex_determination_mammals	19	14	12	73.68	63.16
31 [3]	www.ebi.ac.uk/biomodels/MODEL1504170002	9	7	6	77.78	66.67
32 [3]	www.ebi.ac.uk/biomodels/MODEL1504170001	9	7	6	77.78	66.67
33 [66]	ginsim.org/node/26	12	12	8	100.0	66.67
34 [30]	ginsim.org/node/225	42	37	29	88.1	69.05
35 [67]	ginsim.org/node/227	28	22	20	78.57	71.43
36 [31]	ginsim.org/node/78	40	29	29	72.5	72.5
37 [68]	ginsim.org/node/180	48	38	35	79.17	72.92
38 [45]	ginsim.org/node/173	53	41	39	77.36	73.58
39 [30]	ginsim.org/node/225	128	103	95	80.47	74.22
40 [45]	ginsim.org/node/173	16	12	12	75.0	75.0
41 [69]	ginsim.org/node/39	20	15	15	75.0	75.0
42 [67]	ginsim.org/node/227	33	26	25	78.79	75.76
43 [45]	ginsim.org/node/173	17	13	13	76.47	76.47
44 [70]	www.ebi.ac.uk/biomodels/MODEL1305240000	30	23	23	76.67	76.67
45 [71]	www.ebi.ac.uk/biomodels/MODEL2004040001	56	44	43	78.57	76.79
46 [72]	www.ebi.ac.uk/biomodels/BIOMD0000000562	28	22	22	78.57	78.57
47 [66]	ginsim.org/node/25	39	31	31	79.49	79.49
48 [73]	ginsim.org/node/188	35	31	28	88.57	80.0
49 [74]	ginsim.org/node/69	10	8	8	80.0	80.0
50 [75]	ginsim.org/model/sex_determination_chicken	10	8	8	80.0	80.0
51 [76]	www.ebi.ac.uk/biomodels/MODEL1610060000	31	26	25	83.87	80.65
52 [77]	ginsim.org/node/248	37	32	30	86.49	81.08
53 [66]	ginsim.org/node/29	16	13	13	81.25	81.25
54 [50]	ginsim.org/node/152	11	9	9	81.82	81.82
55 [66]	ginsim.org/node/21	50	41	41	82.0	82.0
56 [30]	ginsim.org/node/225	110	91	91	82.73	82.73
57 [45]	ginsim.org/node/173	18	15	15	83.33	83.33
58 [78]	ginsim.org/node/50	6	6	5	100.0	83.33
59 [79]	ginsim.org/model/HSPC_MSC	26	23	22	88.46	84.62
60 [80]	ginsim.org/node/82	7	6	6	85.71	85.71
61 [80]	ginsim.org/node/82	7	6	6	85.71	85.71
62 [81]	ginsim.org/node/31	14	12	12	85.71	85.71
63 [82]	ginsim.org/node/41	21	18	18	85.71	85.71
64 [76]	www.ebi.ac.uk/biomodels/MODEL1610060001	29	27	25	93.1	86.21
65 [75]	ginsim.org/model/sex_determination_chicken	15	13	13	86.67	86.67
66 [20]	ginsim.org/node/87	60	52	52	86.67	86.67
67 [83]	ginsim.org/node/71	56	50	50	89.29	89.29
68 [84]	ginsim.org/model/SP	19	17	17	89.47	89.47
69 [81]	ginsim.org/node/37	10	9	9	90.0	90.0
70 [81]	ginsim.org/node/37	10	9	9	90.0	90.0
71 [85]	ginsim.org/model/signalling-prostate-cancer	133	121	121	90.98	90.98
72 [86]	ginsim.org/node/229	133	124	122	93.23	91.73
73 [87]	ginsim.org/model/T2DM	26	24	24	92.31	92.31
74 [88]	ginsim.org/node/194	14	13	13	92.86	92.86
75 [86]	ginsim.org/node/229	53	50	50	94.34	94.34
76 [89]	ginsim.org/node/240	18	17	17	94.44	94.44
77 [90]	ginsim.org/node/191	20	19	19	95.0	95.0
78 [91]	ginsim.org/node/234	61	58	58	95.08	95.08
79 [90]	ginsim.org/node/191	32	31	31	96.88	96.88
80 [92]	ginsim.org/node/216	34	33	33	97.06	97.06
81 [80]	ginsim.org/node/82	7	7	7	100.0	100.0
82 [93]	ginsim.org/node/189	14	14	14	100.0	100.0
83 [7]	ginsim.org/node/4	10	10	10	100.0	100.0
84 [93]	ginsim.org/node/189	13	13	13	100.0	100.0
85 [80]	ginsim.org/node/82	7	7	7	100.0	100.0
86 [94]	ginsim.org/model/p53-Mdm2	6	6	6	100.0	100.0

Table S3: Large-scale application of BBE on the repositories GINsim and Biomodels. *Model ID* gives the model identifier as in Fig. 4 and a reference. *URL* allows to download the model from the repositories. *Size* presents the number of variables in the original BN, and in its IS and maximal reduction (N, N_i, N_m resp.). The last column contains the ratios: $R_i = N_i/N$ and $R_m = N_m/N$. In most of the cases, BBE took less than a second. The largest runtime is about 2 seconds.

Model ID	Original		IS Reduced		Max Reduced	
	N	Time (s)	N_i	Time (s)	N_m	Time (s)
1 [50]	26	time-out	7	0.046	4	0.048
2 [51]	19	21.739	11	0.433	3	0.037
3 [50]	24	time-out	11	5.768	4	0.041
4 [52]	57	time-out	34	time-out	11	0.453
6 [53]	18	12.470	13	2.749	4	0.038
8 [50]	16	2.581	8	0.265	5	0.229
9 [50]	23	339.396	13	0.259	8	0.042
10 [50]	19	20.333	8	6.931	7	5.472
11 [50]	24	time-out	11	0.089	9	0.055
12 [55]	37	time-out	16	2.492	14	0.954
13 [50]	34	time-out	23	417.864	15	1.191
14 [50]	18	7.836	11	4.178	8	2.759
15 [56]	30	time-out	25	time-out	14	0.485
16 [54]	24	time-out	15	6.602	12	0.179
21 [50]	18	11.608	11	0.116	10	0.079
22 [61]	36	time-out	23	440.921	21	96.438
30 [55]	19	21.901	14	0.568	12	0.180
35 [67]	28	time-out	22	167.489	20	39.467
40 [45]	16	3.554	12	0.265	12	0.251
41 [69]	20	37.951	15	0.972	15	0.968
43 [45]	17	9.598	13	1.751	13	1.256
44 [70]	30	time-out	23	346.175	23	338.690
46 [72]	28	time-out	22	160.363	22	166.330
53 [66]	16	3.800	13	0.647	13	0.652
57 [45]	18	12.099	15	1.270	15	1.307
59 [79]	26	time-out	23	425.386	22	202.055
62 [81]	14	0.655	12	0.155	12	0.185
63 [82]	21	88.812	18	10.205	18	10.148
65 [75]	15	1.159	13	0.326	13	0.305
68 [84]	19	21.972	17	4.465	17	4.397
74 [88]	14	2.228	13	0.894	13	0.897
76 [89]	18	11.451	17	4.376	17	4.453
77 [90]	20	43.676	19	23.144	19	22.595

Table S4: The table displays the results of the large-scale validation for STG generation. The first column contains the model identifier. Then, each 2-columns block *Original*, *IS Reduced*, and *Max Reduced* contains information on STG generation on the original BN and its IS and maximal reductions, respectively. In particular, N, N_i and N_m contain the number of variables, while *Time* contains the time in seconds, averaged over 3 runs, for STG generation by PyBoolNet.

Model	Original			IS Reduced			Max Reduced		
ID	N	Attractors	Time (s)	N_i	Attractors	Time (s)	N_m	Attractors	Time (s)
1 [50]	26	16384	16.771	7	2	0.002	4	2	0.002
2 [51]	19	2832	2.296	11	16	0.011	3	2	0.002
3 [50]	24	64	0.067	11	2	0.002	4	2	0.002
4 [52]	57	9984	20.124	34	4	0.009	11	2	0.002
5 [53]	10	125	0.072	2	2	0.001	2	2	0.001
6 [53]	18	339	0.283	13	14	0.011	4	2	0.002
7 [54]	8	24	0.030	4	3	0.002	2	2	0.001
8 [50]	16	512	0.367	8	2	0.0026	5	2	0.002
9 [50]	23	512	0.473	13	2	0.003	8	2	0.002
10 [50]	19	4110	3.284	8	2	0.002	7	2	0.002
11 [50]	24	8192	7.866	11	2	0.002	9	2	0.002
12 [55]	37	15459	21.847	16	16	0.014	14	12	0.010
13 [50]	34	1536	1.952	23	3	0.005	15	2	0.003
14 [50]	18	256	0.203	11	2	0.002	8	2	0.002
15 [56]	30	9	0.019	25	3	0.005	14	2	0.003
16 [54]	24	1596	1.572	15	9	0.009	12	4	0.004
17 [57]	92	7360	24.031	72	50	0.144	46	2	0.008
18 [58]	103	–	time-out	54	9	0.145	52	9	0.126
19 [59]	66	77876	235.756	43	2	0.008	35	2	0.005
20 [60]	45	1	0.009	24	1	0.002	24	1	0.003
21 [50]	18	384	0.313	11	3	0.003	10	2	0.002
22 [61]	36	86358	118.792	23	12	0.014	21	12	0.013
23 [61]	71	121976	359.252	44	12	0.025	42	12	0.023
24 [62]	96	4	0.030	81	2	0.015	57	2	0.009
25 [63]	82	1	0.013	49	1	0.005	49	1	0.005
26 [3]	10	10	0.011	7	3	0.002	6	2	0.002
27 [3]	10	10	0.011	7	3	0.002	6	2	0.002
28 [64]	41	990	1.50	27	6	0.010	25	5	0.008
29 [65]	218	–	time-out	140	68	2.433	136	68	2.182
30 [55]	19	308	0.255	14	10	0.009	12	7	0.006
31 [3]	9	7	0.009	7	4	0.003	6	3	0.002
32 [3]	9	7	0.009	7	4	0.003	6	3	0.002
34 [30]	42	2	0.010	37	2	0.005	29	2	0.004
35 [67]	28	97	0.121	22	23	0.025	20	23	0.023
36 [31]	40	8	0.019	29	2	0.005	29	2	0.005
37 [68]	48	22	0.047	38	4	0.009	35	3	0.007
38 [45]	53	40	0.138	41	17	0.038	39	17	0.034
39 [30]	128	8	0.072	103	2	0.0232	95	2	0.021
40 [45]	16	20	0.023	12	5	0.005	12	5	0.004
41 [69]	20	1	0.007	15	1	0.002	15	1	0.002
42 [67]	33	71	0.104	26	29	0.038	25	29	0.035
43 [45]	17	22	0.027	13	4	0.004	13	4	0.004
44 [70]	30	6	0.026	23	6	0.008	23	6	0.008
45 [71]	56	1972	4.250	44	2	0.009	43	2	0.009
46 [72]	28	2	0.009	22	2	0.003	22	2	0.004
47 [66]	39	3	1.387	31	3	0.073	31	3	0.072
48 [73]	35	62	0.102	31	8	0.015	28	8	0.014
49 [74]	10	1	0.005	8	1	0.002	8	1	0.002
50 [75]	10	48	0.032	8	12	0.007	8	12	0.007
51 [76]	31	53	0.072	26	23	0.028	25	17	0.021
52 [77]	37	57	0.136	32	2	0.008	30	2	0.007
53 [66]	16	2	0.008	13	2	0.003	13	2	0.003
54 [50]	11	4	0.007	9	2	0.002	9	2	0.002
55 [66]	50	1	2.343	41	1	0.134	41	1	0.134
56 [30]	110	4	0.030	91	2	0.016	91	2	0.016
57 [45]	18	22	0.027	15	5	0.005	15	5	0.006
59 [79]	26	6	0.017	23	3	0.006	22	3	0.006
60 [80]	7	1	0.005	6	1	0.002	6	1	0.001
61 [80]	7	1	0.005	6	1	0.001	6	1	0.001
62 [81]	14	1	0.006	12	1	0.002	12	1	0.002
63 [82]	21	8	0.014	18	8	0.009	18	8	0.009
64 [76]	29	27	0.096	27	15	0.021	25	8	0.013
65 [75]	15	48	0.041	13	12	0.010	13	12	0.010
66 [20]	60	264	0.676	52	6	0.022	52	6	0.021
67 [83]	56	3	0.034	50	3	0.016	50	3	0.019
68 [84]	19	19	0.024	17	5	0.006	17	5	0.006
69 [81]	10	13	0.014	9	5	0.004	9	5	0.004
70 [81]	10	2	0.006	9	2	0.002	9	2	0.002
71 [85]	133	7008	108.475	121	4	0.103	121	4	0.102
72 [86]	133	11	2.882	124	5	2.118	122	5	2.161
73 [87]	26	32	0.045	24	8	0.012	24	8	0.012
74 [88]	14	4	0.009	13	2	0.003	13	2	0.003
75 [86]	53	10	0.502	50	5	0.220	50	5	0.217
76 [89]	18	5	0.012	17	5	0.006	17	5	0.006
77 [90]	20	13	0.020	19	7	0.008	19	7	0.008
78 [91]	61	32	0.115	58	7	0.034	58	7	0.033
79 [90]	32	15	0.032	31	9	0.018	31	9	0.018
80 [92]	34	20	0.048	33	10	0.031	33	10	0.030

Table S5: The table contains the large-scale validation for the attractor analysis. The first column is the model identifier and its reference. Each 3-columns block *Original*, *IS Reduced*, and *Max Reduced* contains information on attractor computation on the original BN and its IS and maximal reductions, respectively. In particular, N, N_i, and N_m contain the number of variables, while *Attractors* contains the number of attractors computed by tool BNS, and *Time* the time in seconds to obtain them averaged over 3 runs.

C PAPER III: Minimization of Dynamical Systems over Monoids

Submitted to the Thirty-Eighth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2023)

Minimization of Dynamical Systems over Monoids

Authors omitted to adhere to double-blind requirements

***Abstract*—Quantitative notions of bisimulation are well-known tools for the minimization of dynamical models such as Markov chains and ordinary differential equations (ODEs). In *forward bisimulations*, each state in the quotient model represents an equivalence class and the dynamical evolution gives the overall sum of its members in the original model. Here we introduce generalized forward bisimulation (GFB) for dynamical systems over commutative monoids and develop a partition refinement algorithm to compute the largest one. When the monoid is $(\mathbb{R}, +)$, we recover probabilistic bisimulation for Markov chains and more recent forward bisimulations for nonlinear ODEs. Using $(\mathbb{R}, \cdot)$ we get nonlinear reductions for discrete-time dynamical systems and ODEs where each variable in the quotient model represents the product of original variables in the equivalence class. When the domain is a finite set such as the Booleans $\mathbb{B}$, we can apply GFB to Boolean networks, a widely used dynamical model in computational biology. Using a prototype implementation of our minimization algorithm for GFB, we find disjunction- and conjunction-preserving reductions on 60 Boolean networks from two well-known repositories, and demonstrate the obtained analysis speed-ups. We also provide the biological interpretation of the reduction obtained for two selected Boolean networks, and we show how GFB enables the analysis of a large one that could not be analyzed otherwise. Using a randomized version of our algorithm we find product-preserving (therefore non-linear) reductions on 21 dynamical weighted networks from the literature that could not be handled by the exact algorithm.**

I. Introduction

Bisimulation is a fundamental tool in computer science for abstraction and minimization, relating models by useful logical and dynamical properties [1]. Originally developed to reason about concurrent processes in a non-quantitative setting [2], it has been extended to quantitative models based on labeled transition systems, such as, e.g., the notion of probabilistic bisimulation [3], closely related to ordinary lumpability for Markov chains [4].

Forward bisimulations relate states based on criteria that depend on their *outgoing* transitions (as opposed to *backward* bisimulations that depend on *incoming* ones, e.g., [5]). When applied to a dynamical system (DS), forward bisimulations preserve properties related to sums of values of state variables. For example, probabilistic bisimulation yields a quotient Markov process where each state represents an equivalence class preserving the sum of the probabilities of its members; forward bisimulation for reaction networks identifies equivalence classes among the chemical species that preserve the total concentration [6], [7]; forward differential equivalence (FDE) for nonlinear ordinary differential equations (ODEs) relates variables preserving sums of their solutions [8].

An attractive feature of bisimulation is that one can compute the largest bisimulation using partition refinement, based on the pioneering solution for concurrent processes [9]. Partition refinement algorithms start from an *initial partition* of variables which is iteratively refined (i.e., its blocks get split) until the obtained partition is a bisimulation. Notably, such notion of initial partition is particularly useful to the modeler to tune the reduction. For example, one can *separate* groups of variables according to given criteria so as to *refine* them, i.e., to prevent that variables from different groups will be aggregated together. This makes bisimulation an effective approach for the minimization of complex DS, adding to cross-disciplinary methods originated in e.g., chemical engineering [10], control theory [11], and systems biology [12].

Thus far, one can identify two common properties of existing forward bisimulations for DS: they preserve sums of state values, and the DS variables take real $\mathbb{R}$ values. There are, however, motivations that call for generalizations of this setting. A forward bisimulation for ODEs can be seen as a special case of *linear lumping* [10], a minimization achieved by a linear projection of the state space operated by a matrix that encodes the partition of the state variables. However, one may be also interested in *nonlinear lumpings* where each state in the reduced model represents a nonlinear transformation of original variables [13].

Another motivating question tackled in this paper is the generalization of the domain on which the DS evolves. Forward bisimulation is not currently applicable to DS that evolve over finite domains. Consider, e.g., the DS

$$\begin{aligned} x_1(k+1) &= x_2(k) \vee x_3(k) \\ x_2(k+1) &= x_1(k) \vee x_3(k) \\ x_3(k+1) &= \neg x_3(k) \wedge (x_1(k) \vee x_2(k)) \end{aligned} \tag{1}$$

where the state variables x_1, x_2, and x_3 are defined over the Booleans $\mathbb{B} = \{0, 1\}$, and k denotes discrete time. This is a Boolean network (BN), an established model of biological systems [14].

Here we develop a more abstract notion of forward bisimulation, *generalized forward bisimulation* (GFB), for a DS over a (commutative) monoid. We show that this is a conservative extension with respect to the literature because we recover available notions of forward bisimulation for DS when the monoid is $(\mathbb{R}, +)$. However, it is more general. For example, over the monoid $(\mathbb{B}, \vee)$ one can prove that variables x_1 and x_2 in (1) are *GFB equivalent*, i.e., we can rewrite the model in terms of $x_1 \vee x_2$ and x_3. Indeed, by computing the disjunction of the left- and right-hand-side of x_1 and x_2 in (1) we get

$$\begin{aligned} x_1(k+1) \vee x_2(k+1) &= x_2(k) \vee x_3(k) \vee x_1(k) \vee x_3(k) \\ &= x_3(k) \vee (x_1(k) \vee x_2(k)). \end{aligned}$$

By using the derived variable $x_{1,2} \equiv x_1 \vee x_2$, we get the *GFB-reduced model*

$$\begin{aligned} x_{1,2}(k+1) &= x_3(k) \vee x_{1,2}(k) \\ x_3(k+1) &= \neg x_3(k) \wedge x_{1,2}(k). \end{aligned} \tag{2}$$

This can be used in place of the original model if one is not interested in the individual values of x_1 and x_2, but only in their disjunction.

Here we show that GFB satisfies desirable properties for bisimulations.

1) Over any commutative monoid $(\mathbb{M}, \oplus)$, GFB characterizes $\oplus$-preserving reductions, in the sense that any DS with fewer state variables which coincides with $\oplus$-operations of original state variables must necessarily be the quotient of a GFB. This generalizes characterization results for Markov chains [3], chemical reaction networks [15], and nonlinear ODEs [8]. Notably, our characterization result also covers the asymptotic dynamics, often of interest when analyzing DS (see, e.g., [16]). We show that GFB preserves all *attractors*, i.e., the states towards which the DS tends to evolve and remain.
2) GFB can be computed by a partition refinement algorithm. We develop a *template* algorithm which hinges on the computation of a formula whose decidability and complexity depend on the domain and the right-hand sides of the dynamical system under study. In general, this can be undecidable. However, when the monoid is $(\mathbb{R}, +)$ our algorithm reduces to that for forward differential equivalence for nonlinear ODEs [8]. Instead, when the domain is $\mathbb{B}$, the problem corresponds to Boolean satisfiability.
3) For polynomial ODEs and the monoid $(\mathbb{R}, \cdot)$, we obtain, to the best of our knowledge, the first algorithm for nonlinear model reduction in (randomized) polynomial time.
4) GFB is effective in practice, both in terms of reduction power and of obtained analysis speed-ups.

Previous results are essentially agnostic to whether the time evolution of the DS is continuous or discrete. More specifically, the criteria for probabilistic bisimulation [3] are the same for both continuous-time and discrete-time Markov chains. Similarly, FDE equivalently applies to both a nonlinear ODE system in the form $\partial_t x = f(x)$ (where ∂_t denotes time derivative) and to a discrete-time nonlinear DS in the form $x(k+1) = f(x(k))$. With GFB, instead, more care has to be taken because this verbatim correspondence does not hold any longer. For this reason, we first develop GFB for discrete-time DS. Then, we consider continuous time by studying GFB for DS over the reals relating to, and extending, results for ODEs.

Applications. Using a prototype implementation, we apply GFB to case studies from different domains. We consider Boolean and multi-valued networks [14], [17], where the latter allows for finer degrees of activation than just 0/1 as in (1). These models are known to suffer from state-space explosion, making model reduction appealing (see, e.g., [18]). We select two case studies from the literature to showcase the physical intelligibility of GFB reductions, and one to show how GFB can enable the analysis of BNs that otherwise could not be analyzed. In the three case studies, we show how initial partitions can be devised using domain knowledge from specific case studies. For example, we show how $(\mathbb{B},\wedge)$ allows to identify and abstract away from distinct *sub-models* (biological pathways); we show how finite monoids and operations $\min$ and $\max$ allow studying *full model (de)activation*, meaning that we obtain reductions that track groups of components whose activation status denote the (de)activation of different mechanisms of the model. We also perform a large-scale validation of GFB on 60 Boolean and multi-valued networks from established repositories (GinSim [19], BioModelsDB [20]), showing how *default* initial partitions can be synthesized automatically in this setting. We show that GFB is *useful* due to its high reduction power, and the high speed-up obtained in attractors computation. We also consider real-valued DS. We study a case study of a higher-order Lotka-Volterra model [21], and we perform a large-scale validation on 72 weighted networks from the Netzschleuder repository [22].

II. Related work

Most of the literature about model minimization can be found for DS over the reals. In this context, the general framework of exact lumping considers reductions by means of both linear and nonlinear operators [23], [24]. The aforementioned notions of bisimulation for Markov chains and FDE can be seen as specific linear reductions that are induced by a partition of the state space. Indeed, this corresponds to a specific type of minimization known as *proper lumping*, where each original variable is represented by only one variable in the reduced model [10]. Since also GFB is developed in the same style, it too can be seen as a special case of exact lumping. However, the largest GFB can be computed in randomized polynomial time when the dynamics is described by polynomials over the monoids $(\mathbb{R}, +)$ or $(\mathbb{R}, \cdot)$, see [25] and Section V. Instead, the computation of exact lumpings hinges, in the case of polynomial dynamics, on symbolic computations with worst-case exponential complexity [13, Section 2.2].

Relying on polynomial invariants [26], [27], $\mathcal{L}$-bisimulation [28], [29] can be seen as a generalization of backward differential equivalence (BDE) [8], a backward-type bisimulation for non-linear ODEs, and is thus complementary to FDE (hence, GFB), as discussed in [28]–[30]. It is also worth noting that neither BDE nor $\mathcal{L}$-bisimulation allow for model reduction through nonlinear transformations, in contrast to GFB. Similarly to $\mathcal{L}$-bisimulation, consistent abstraction (aka bisimulation) [31]–[33] is complementary to GFB. Indeed, for a so-called observation function, the largest consistent abstraction gives rise to a minimal reduced DS which coincides with the original one up to the chosen observation function. Instead, computing the largest GFB corresponds to finding an observation function which induces a largest consistent abstraction. Hence, GFB reduces across observation functions, while consistent abstraction reduces with respect to a given observation function. Moreover,

in contrast to consistent abstraction, GFB considers the subclass of observation functions induced by equivalence relations. To the best of our knowledge, the computation of an observation function yielding a minimal reduced model has been investigated for linear dynamics only [31].

Reduction techniques exist for BNs. Boolean backward equivalence (BBE) is a backward-type bisimulation [18], in line exact lumpability for Markov chains [4] and BDE. Hence, it can be shown that BBE and GFB (applied to BNs) are not comparable. Other approaches for BN reduction are based on *variable absorption* (e.g., [34], [35]) where selected variables are *absorbed* by the update functions of their target variables by replacing all occurrences of the absorbed variables with their update functions. These approaches are complementary to GFB because they do not compute exact reductions.

III. Preliminaries

In this section we formalize the notion of discrete-time DS and of attractor for discrete-time DS [36], and notation considered in this paper. Then, we provide a running example used throughout the text.

Definition 1 (Dynamical System)**.** *A discrete-time DS is a pair $D = (X, F)$ where $X = \{x_1, \dots, x_n\}$ are variables and $F = \{f_{x_1}, \dots, f_{x_n}\}$ is a set of update functions, where $f_{x_i} : \mathbb{M}^X \to \mathbb{M}$ is the update function of variable x_i. Elements of $\mathbb{M}^X$ are* states*. The solution (simulation) of D for initial state $s(0) \in \mathbb{M}^X$ is given by the sequence $(s(k))_{k\geq 0}$, where $s(k+1) = F(s(k))$ for all $k \geq 0$.*

We use R to denote an equivalence relation over X, and $\mathcal{X}_R$ the induced partition. We often do not distinguish among an equivalence relation and its induced partition. If not mentioned, we assume that $\oplus : \mathbb{M} \times \mathbb{M} \to \mathbb{M}$ is such that $(\mathbb{M}, \oplus)$ is a commutative monoid with neutral element $0_\oplus$. Moreover, G^I denotes the set of all (total) functions from I to G and $f[a/b]$ is the term arising by replacing each occurrence of a by b in f.

As running example we use a BN from [37] that describes cell differentiation. Deeper biological interpretation and its reduction will be given in Section VII.

Example 1. *Let (X, F) be a discrete-time DS with Boolean variables $X = \{\mathsf{SCR}, \mathsf{SHR}, \mathsf{JKD}, \mathsf{MGP}, \mathsf{WOX5}, \mathsf{CLEX}, \mathsf{PLT}, \mathsf{ARF}, \mathsf{AUXIAA}, \mathsf{AUXIN}\}$ and update function $F : \mathbb{B}^X \to \mathbb{B}^X$ with*

$$f_{\mathsf{SCR}} = \mathsf{SHR} \wedge \mathsf{SCR} \wedge (\mathsf{JKD} \vee \neg\mathsf{MGP}) \qquad f_{\mathsf{CLEX}} = \mathsf{SHR} \wedge \mathsf{CLEX}$$
$$f_{\mathsf{SHR}} = \mathsf{SHR} \qquad f_{\mathsf{PLT}} = \mathsf{ARF}$$
$$f_{\mathsf{JKD}} = \mathsf{SHR} \wedge \mathsf{SCR} \qquad f_{\mathsf{ARF}} = \neg\mathsf{AUXIAA}$$
$$f_{\mathsf{MGP}} = \mathsf{SHR} \wedge \mathsf{SCR} \wedge \neg\mathsf{WOX5} \qquad f_{\mathsf{AUXIAA}} = \neg\mathsf{AUXIN}$$
$$f_{\mathsf{WOX5}} = \mathsf{ARF} \wedge \mathsf{SHR} \wedge \mathsf{SCR} \wedge \neg\mathsf{CLEX} \qquad f_{\mathsf{AUXIN}} = \mathsf{AUXIN}$$

Monoids for the DS are $(\mathbb{B}, \oplus)$, $\oplus \in \{\wedge, \vee, XOR\}$, with neutral elements $1, 0, 0$.

Definition 2 (Attractor)**.** *Let $D = (X, F)$ be a discrete-time DS. A non-empty set $A \subseteq \mathbb{M}^X$ is called attractor of D (wrt some given topology of $\mathbb{M}^X$) whenever*

- *A is invariant under F, that is, $F(A) \subseteq A$;*
- *there is an open neighborhood B of A s.t. for any $v \in B$ there exists a $\nu \geq 1$ such that $F^n(v) \in A$ for all $n \geq \nu$. B is called a* basin of attraction *of A.*

Example 2. *Let $s = (0,0,0,0,0,1,1,1,0,1) \in \mathbb{B}^X$ denote a state of the DS from Example 1 where only the variables $\mathsf{CLEX}, \mathsf{PLT}, \mathsf{ARF}, \mathsf{AUXIN}$ are active. By applying the update functions we get $F(s) = s' = (0,0,0,0,0,0,1,1,0,1) \in \mathbb{B}^X$, where PLT, ARF and AUXIN are active. If we apply the update functions again, the system remains in the same state, i.e., $F(s') = s'$, meaning that $\{s'\}$ is an attractor.*

IV. Generalized Forward Bisimulation

Here we define generalized forward bisimulation (GFB), the notion of GFB reduction, and show that GFB reductions preserve the original model dynamics.

Definition 3 (Generalized Forward Bisimulation)**.** *Let $D = (X, F)$ be a discrete-time DS, $(\mathbb{M}, \oplus)$ a commutative monoid and $\mathcal{X}_R$ a partition of X. Then, $\mathcal{X}_R$ is a GFB when the following formula holds true:*

$$\forall s, s' \in \mathbb{M}^X. \bigwedge_{C \in \mathcal{X}_R} \Big(\bigoplus_{x_i \in C} s_{x_i} = \bigoplus_{x_i \in C} s'_{x_i} \Big) \implies \bigwedge_{C \in \mathcal{X}_R} \Big(\bigoplus_{x_i \in C} f_{x_i}(s) = \bigoplus_{x_i \in C} f_{x_i}(s') \Big).$$

The homomorphism of R, denoted by $\psi_R : \mathbb{M}^X \to \mathbb{M}^{\mathcal{X}_R}$, is given by

$$\psi_R(s)_C = \bigoplus_{x_i \in C} s_{x_i}, \quad \text{for all } C \in \mathcal{X}_R.$$

Example 3. *For $\oplus = \wedge$, $\mathcal{X}_R = \{C, \{\mathsf{PLT}\}, \{\mathsf{ARF}\}, \{\mathsf{AUXIAA}\}, \{\mathsf{AUXIN}\}\}$ is a GFB for our running example, where $C = \{\mathsf{SCR}, \mathsf{SHR}, \mathsf{JKD}, \mathsf{MGP}, \mathsf{WOX5}, \mathsf{CLEX}\}$. This means that the running example can be rewritten solely in terms of conjunctions over all variables in C, and the other individual variables. To this end, we first note that for all $x_i \notin C$ we have that f_{x_i} is independent of any $x_j \in C$. [1] Moreover, the update functions of $\mathsf{WOX5}$ and CLEX contain terms $\neg\mathsf{CLEX}$ and CLEX, respectively, therefore the conjunction of their update functions (and of all variables in C) can be simply rewritten as 0 since: $\bigwedge_{x_i \in C} f_{x_i}(s) = s_{\mathsf{CLEX}} \wedge \neg s_{\mathsf{CLEX}} \wedge (\dots) = 0$.*

Definition 4 (Reduced DS)**.** *The reduction D/R of a discrete-time DS $D = (X, F)$ for an equivalence R, is the DS $(\mathcal{X}_R, F_R)$ with $F_R = (f_C)_{C \in \mathcal{X}_R}$ such that*

$$f_C = \bigoplus_{x_i \in C} f_{x_i}[x_k / 0_\oplus : x_k \notin \hat{X}][x_{i_{C'}} / x_{C'} : C' \in \mathcal{X}_R],$$

where $x_{i_C} \in C$ is a representative of $C \in \mathcal{X}_R$ and $\hat{X} = \{x_{i_C} : C \in \mathcal{X}_R\}$ is the set of all representatives.

[1] However, the original system is not trivially decoupled in variables in C and variables not in C, because ARF appears in the update function of $\mathsf{WOX5}$.

Example 4. *We compute the reduced DS of our running example for the GFB $\mathcal{X}_R$ from Example 3. We choose* JKD *as representative of C, while the representative for the other (singleton) blocks is obvious. With this, we obtain*

$$\begin{aligned} f_C &= \bigwedge_{x_k \in C} f_{x_k}[x_k/1 : x_k \notin \hat{X}][x_{i_{C'}}/x_{C'} : C' \in \mathcal{X}_R] \\ &= 1 \wedge 1 \wedge \big(C \vee \neg 1\big) \wedge 1 \wedge 1 \wedge 1 \wedge 1 \\ &\quad \wedge 1 \wedge \neg 1 \wedge \{\mathsf{ARF}\} \wedge 1 \wedge 1 \wedge \neg 1 \wedge 1 \wedge 1 = 0 \end{aligned}$$

For all other blocks, instead, we obtain

$$\begin{aligned} f_{\{\mathsf{PLT}\}} &= \{\mathsf{ARF}\}, & f_{\{\mathsf{ARF}\}} &= \neg\{\mathsf{AUXIAA}\}, \\ f_{\{\mathsf{AUXIAA}\}} &= \neg\{\mathsf{AUXIN}\}, & f_{\{\mathsf{AUXIN}\}} &= \{\mathsf{AUXIN}\} \end{aligned}$$

Remark 1. *We note that, syntactically, the reduced DS depends on the choice of representatives. However, if R is a GFB, then Theorem 1 guarantees that such choice does not affect the* semantics *of the reduced DS.*

We now show that D and D/R have *same dynamics up to* ψ_R iff R is a GFB.

Theorem 1 (GFB characterization via model dynamics)**.** *Fix a DS $D = (X, F)$, a partition $\mathcal{X}_R$ of X, $D/R = (\mathcal{X}_R, F_R)$, and a commutative monoid $(\mathbb{M}, \oplus)$. Then, R is a GFB iff for any initial state $s_0 \in \mathbb{M}^X$ the solutions of D and D/R for s_0 and $\hat{s}_0 = \psi_R(s_0)$, respectively, are equal up to ψ_R. That is:*

$$\hat{s}_k = \psi_R(s_k), \quad \textit{for } k \geq 0,$$

where $s_{k+1} = F(s_k)$ and $\hat{s}_{k+1} = F_R(\hat{s}_k)$.

Proof of Theorem 1. Let R be a GFB, pick $s_0 \in \mathbb{M}^X$ and set $\hat{s}_0 = \psi_R(s_0) \in \mathbb{M}^{\mathcal{X}_R}$. We next show that $\hat{s}_k = \psi_R(s_k)$ by induction over $k \geq 0$. Since the base case $k = 0$ is true by construction, we can turn to the induction step. For $k \geq 0$, we obtain

$$\hat{s}_{k+1} = F_R(\hat{s}_k) = F_R(\psi_R(s_k)) = \psi_R(F(s_k)) = \psi_R(s_{k+1}),$$

where the second identity follows from the induction hypothesis, while the third identity follows from the definition of F_R and the fact that R is a GFB. Conversely, if $\hat{s}_k = \psi_R(s_k)$ for all $k \geq 0$, we can conclude for $k = 0$ and arbitrary $s_0 \in \mathbb{M}^X$ that

$$\psi_R(F(s_0)) = \psi_R(s_1) = \hat{s}_1 = F_R(\hat{s}_0) = F_R(\psi_R(s_0)),$$

thus showing that R is a GFB. □

Theorem 1 readily implies the following result on attractors.

Corollary 1. *Let $D = (X, F)$ be a DS, $(\mathbb{M}, \oplus)$ a commutative monoid, R a GFB and $D/R = (\mathcal{X}_R, F_R)$. Then, we have the following two (equivalent) statements.*

- *If $A \subseteq \mathbb{M}^X$ is an attractor of D, then $\psi_R(A) \subseteq \mathbb{M}^{\mathcal{X}_R}$ is an attractor of D/R.*
- *If $A \subseteq \mathbb{M}^{\mathcal{X}_R}$ is not an attractor of D/R, then $\psi_R^{-1}(A) \subseteq \mathbb{M}^X$ is not an attractor of D.*

Example 5. *We consider the attractor $s' = \{(0,0,0,0,0,0,1,1,0,1)\}$ from Example 2. The homomorphism ψ_R maps the attractor to $\psi_R(s') = \big\{(0,1,1,0,1)\big\}$. Corollary 1 ensures that the set $\psi_R(s')$ is an attractor of the reduced system D/R. Indeed, by applying the update functions F_R to $(0,1,1,0,1)$, the reduced system remains at the same state, and thus $\psi_R(s')$ is invariant under F_R.*

V. Computation of the largest GFB

Computing the largest (or coarsest) GFB that refines a given initial partition is based on the classic partition refinement algorithm [9] where the blocks of an initial partition are iteratively refined (or split) until a GFB is obtained. The largest GFB is obtained when the initial partition contains one block only. Different initial partitions can be useful to tune reductions to preserve variables of interest (see, e.g., Section VII). Here we prove that there exists a unique largest GFB that refines a given initial partition, and that the algorithm computes it.

Theorem 2. *Let $D = (X, F)$ be a discrete-time DS, and $\mathcal{X}_R$ a partition of X. There exists a unique coarsest GFB $\mathcal{H}$ that refines $\mathcal{X}_R$.*

Proof of Theorem 2. Fix arbitrary GFBs $\sim_1, \ldots, \sim_\nu \subseteq R$ and let $\mathcal{H}_1, \ldots, \mathcal{H}_\nu$ be the corresponding partitions, i.e., $\mathcal{H}_i = X_{\sim_i}$. Moreover, let $\sim_* := \big(\bigcup_{i=1}^m \sim_i\big)^*$ and $\mathcal{H}^* := X_{\sim_*}$, where the asterisk denotes transitive closure of a relation. At last, let $x_{i_{H^*}} \in H^*$ denote some representative of $H^* \in \mathcal{H}^*$. With this, pick an arbitrary $H^* \in \mathcal{H}^*$. By construction of $\mathcal{H}^*$, there exist $x_0, \ldots, x_k \in X$ and $i_0, \ldots, i_{k-1} \in \{1, \ldots, \nu\}$ so that $\{x_0, \ldots, x_k\} = H^*$, $x_k = x_{i_{H^*}}$ and $x_j \sim_{i_j} x_{j+1}$ for all $0 \leq j \leq k-1$. Moreover, for any $G^* \in \mathcal{H}^*$ and $1 \leq i \leq \nu$, there exist (unique) $G_1^i, \ldots, G_{m_i}^i \in \mathcal{H}_i$ such that $\biguplus_{l=1}^{m_i} G_l^i = G^*$. Since $x_j \sim_{i_j} x_{j+1}$ and $\mathcal{H}_{i_j}$ is a GFB, we obtain

$$\begin{aligned} \bigoplus_{x_\iota \in G^*} f_{x_\iota} &= \bigoplus_{l=1}^{m_{i_j}} \bigoplus_{x_\iota \in G_l^{i_j}} f_{x_\iota} \\ &= \bigoplus_{l=1}^{m_{i_j}} \bigoplus_{x_\iota \in G_l^{i_j}} f_{x_\iota}[x_j/0_\oplus][x_{j+1}/(x_j \oplus x_{j+1})] \\ &= \bigoplus_{x_\iota \in G^*} f_{x_\iota}[x_j/0_\oplus][x_{j+1}/(x_j \oplus x_{j+1})] \end{aligned}$$

Since $\{x_0, x_1, \ldots, x_k\} = H^*$ and $x_k = x_{i_{H^*}}$, an application of the argument for all $0 \leq j \leq k-1$ implies that $\bigoplus_{x_\iota \in G^*} f_{x_\iota}$ is equivalent to

$$\bigoplus_{x_\iota \in G^*} f_{x_\iota}[x_k/0_\oplus : x_k \in H^*, x_k \neq x_{i_{H^*}}][x_{i_{H^*}}/\bigoplus_{x_l \in H^*} x_l]$$

Since the choice of $G^*, H^* \in \mathcal{H}^*$ was arbitrary, we infer that $\mathcal{H}^*$ is a GFB. □

A partition refinement algorithm for computing GFB needs a condition to tell: (i) if the current partition is a GFB, and, if not, (ii) how to split its blocks towards getting a GFB. Definition 3 can only be used for Point (i). Theorem 3 below provides a binary, relation-driven, characterization of GFB

which allows for Point (ii). The intuition is that, by applying such binary characterization pairwise to all variables in each block of the current partition, we get the sub-blocks in which they should be split in the next iteration.

Theorem 3 (Binary Characterization of GFB)**.** *Let $D = (X, F)$ be a DS, $(\mathbb{M}, \oplus)$ a commutative monoid, R an equivalence relation on X, and $\mathcal{X}_R$ the induced partition. Then, $\mathcal{X}_R$ is a GFB if and only if for any $(x_i, x_j) \in R$ with $x_i \neq x_j$, the following formula holds (where $0_\oplus$ is the neutral element of $\oplus$):*

$$\Psi^{\mathcal{X}_R}_{x_i,x_j} \equiv \bigwedge_{C\in\mathcal{X}_R} \Big(\bigoplus_{x_k\in C} f_{x_k} = \bigoplus_{x_k\in C} f_{x_k}[x_i/0_\oplus][x_j/(x_i \oplus x_j)] \Big)$$

Proof of Theorem 3. Let us assume first that $\mathcal{X}_R$ is a GFB, pick an arbitrary $(x_i, x_j) \in R$ and pick the unique $C' \in \mathcal{X}_R$ such that $x_i, x_j \in C'$. With this, define $s' := s[x_i \mapsto 0_\oplus][x_j \mapsto s_{x_i} \oplus s_{x_j}]$ for an arbitrary $s \in \mathbb{M}^X$, where $s[x_k \mapsto b]_{x_k} = b$ and $s[x_k \mapsto b]_{x_l} = s_{x_l}$ for all $b \in \mathbb{M}$ and $x_l \neq x_k$. Then, since $\oplus$ is commutative and associative and because $\mathcal{X}_R$ is a GFB, we have that

$$\bigwedge_{C\in\mathcal{X}_R} \Big(\bigoplus_{x_i\in C} f_{x_i}(s) = \bigoplus_{x_i\in C} f_{x_i}(s') \Big). \tag{3}$$

Since the choice of $(x_i, x_j) \in R$ and $s \in \mathbb{M}^X$ was arbitrary, we infer that $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ is valid. For the converse, let us assume that $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ holds true for all $(x_i, x_j) \in R$ and pick any two $s, s' \in \mathbb{M}^X$ such that

$$\bigwedge_{C\in\mathcal{X}_R} \Big(\bigoplus_{x_i\in C} s_{x_i} = \bigoplus_{x_i\in C} s'_{x_i} \Big) \tag{4}$$

With this, pick for any $C \in \mathcal{X}_R$ some arbitrary representative $x_{i_C} \in C$ and let $\hat{X} = \{x_{i_C} : C \in \mathcal{X}_R\}$ be the set of all representatives. For any $(x_i, x_j) \in R$, define $s_{i\to j} := s[x_i \mapsto 0_\oplus, x_j \mapsto s_{x_i} \oplus s_{x_j}]$. With this, the fact that $\oplus$ is commutative and associative ensures the existence of a sequence $x_{i_1}, x_{i_2}, ..., x_{i_k}$ for which $\hat{s} = (((s_{i_1\to i_2})_{i_2\to i_3}) \ldots)_{i_{k-1}\to i_k}$ is such that

$$\bigwedge_{C\in\mathcal{X}_R} \Big(\bigoplus_{x_i\in C} s_{x_i} = \bigoplus_{x_i\in C} \hat{s}_{x_i} \Big),$$

$\hat{s}_{x_i} = 0_\oplus$ for all $x_i \notin \hat{X}$ and $\hat{s}_{x_{i_C}} = \bigoplus_{x_i\in C} s_{x_i}$ for all $C \in \mathcal{X}_R$. Since $\Psi^{\mathcal{X}_R}_{x_{i_l},x_{i_{l+1}}}$ is valid for all $1 \leq l \leq k-1$, we obtain

$$\bigwedge_{C\in\mathcal{X}_R} \Big(\bigoplus_{x_i\in C} f_{x_i}(s) = \bigoplus_{x_i\in C} f_{x_i}(\hat{s}) \Big).$$

A similar argument for s' ensures that there is an $\hat{s}'$ such that $\hat{s}'_{x_i} = 0_\oplus$ for all $x_i \notin \hat{X}$, $\hat{s}'_{x_{i_C}} = \bigoplus_{x_i\in C} s'_{x_i}$ for all $C \in \mathcal{X}_R$ and

$$\bigwedge_{C\in\mathcal{X}_R} \Big(\bigoplus_{x_i\in C} s'_{x_i} = \bigoplus_{x_i\in C} \hat{s}'_{x_i} \Big),$$
$$\bigwedge_{C\in\mathcal{X}_R} \Big(\bigoplus_{x_i\in C} f_{x_i}(s') = \bigoplus_{x_i\in C} f_{x_i}(\hat{s}') \Big).$$

Algorithm 1: Compute the largest GFB that refines an initial partition $\mathcal{H}$ for DS (X, F).

```
while true do
    H' ← ∅
    for all H ∈ H do
        R ← {(x_i, x_j) ∈ H × H : if x_i ≠ x_j, then Ψ^H_{x_i,x_j} and Ψ^H_{x_j,x_i}}
        H' ← H' ∪ (H/R)
    end for
    if H = H' then
        return H
    else
        H ← H'
    end if
end while
```

Thanks to (4), we infer that $\hat{s} = \hat{s}'$. This, in turn, implies the desired relation (3), thus showing that $\mathcal{X}_R$ is a GFB if and only if $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ is valid for all $(x_i, x_j) \in R$. □

The binary characterization tells us that we can rewrite an $\oplus$-expression of the update functions of a block of a GFB in terms of $\oplus$-expressions of pairs of GFB equivalent variables x_i and x_j. This can be done by successively moving, pair by pair, all variables of a GFB equivalence class to a chosen representative.

Example 6. *Let us consider the GFB $\mathcal{X}_R$ from Example 3, the only non-singleton block $C \in \mathcal{X}_R$, and the variables* SHR, JKD $\in C$. *With $\oplus = \wedge$ and $0_\wedge = 1$, we obtain*

$$\begin{aligned}
\bigwedge_{x_k\in C} f_{x_k} &= \mathsf{SHR} \wedge \mathsf{SCR} \wedge (\mathsf{JKD} \vee \neg\mathsf{MGP}) \\
&\quad \wedge \mathsf{SHR} \wedge \mathsf{SHR} \wedge \mathsf{SCR} \wedge \mathsf{SHR} \wedge \mathsf{SCR} \wedge \neg\mathsf{WOX5} \wedge \mathsf{ARF} \wedge \\
&\quad \mathsf{SHR} \wedge \mathsf{SCR} \wedge \neg\mathsf{CLEX} \wedge \mathsf{SHR} \wedge \mathsf{CLEX} \\
&= 0 \\
&= 1 \wedge \mathsf{SCR} \wedge ((\mathsf{JKD} \wedge \mathsf{SHR}) \vee \neg\mathsf{MGP}) \\
&\quad \wedge 1 \wedge 1 \wedge \mathsf{SCR} \wedge 1 \wedge \mathsf{SCR} \wedge \neg\mathsf{WOX5} \wedge \mathsf{ARF} \\
&\quad \wedge 1 \wedge \mathsf{SCR} \wedge \neg\mathsf{CLEX} \wedge 1 \wedge \mathsf{CLEX} \\
&= \bigwedge_{x_k\in C} f_{x_k}[\mathsf{SHR}/1, \mathsf{JKD}/(\mathsf{SHR} \wedge \mathsf{JKD})]
\end{aligned}$$

For any other block the clause is trivially true because SHR *and* JKD *appear only in the update functions of variables in C. Hence, $\Psi^{\mathcal{X}_R}_{\mathsf{SHR},\mathsf{JKD}}$ is valid. Similarly, we can show that $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ is valid for all $(x_i, x_j) \in R, x_i \neq x_j$. Hence $\mathcal{X}_R$ is a GFB.*

The next result addresses the algorithmic computation of the largest GFB.

Theorem 4. *Let $D = (X, F)$ be a discrete-time DS and X_R a partition. Algorithm 1 computes the largest GFB refining R by deciding at most $\mathcal{O}(|X|^3)$ instances of formula $\Psi^{\mathcal{H}}_{x_i,x_j}$. If $\mathbb{M}$ is finite, any formula $\Psi^{\mathcal{H}}_{x_i,x_j}$ is decidable.*

Proof of Theorem 4. Pick the largest (i.e., coarsest) GFB $\mathcal{H}_*$ that refines X_R using Theorem 2. With this, set $\mathcal{H}_0 := X_R$ and define for all $k \geq 0$ and $H \in \mathcal{H}_k$

$$R_k(H) := \{(x_i, x_j) \in H \times H : x_i \neq x_j \Rightarrow \Psi^{\mathcal{H}_k}_{x_i,x_j} \wedge \Psi^{\mathcal{H}_k}_{x_j,x_i}\}$$
$$\mathcal{H}_{k+1} := \bigcup_{H \in \mathcal{H}_k} H/R^*_k(H),$$

where $R^*_k(H)$ denotes the transitive closure of $R_k(H)$. By construction, $R_k(H)$ is reflexive and symmetric, thus implying $\bigoplus_{x_i \in H} f_{x_i}(s) = \bigoplus_{x_i \in H} f_{x_i}(\tilde{s})$ for all $s \in \mathbb{M}^X$, $H \in \mathcal{H}_k$, where

$$\tilde{s} = s[x_j \mapsto 0_\oplus : x_j \notin \hat{X}_{k+1}][x_{i_{C'}} \mapsto \bigoplus_{x_j \in C'} s_{x_j} : C' \in \mathcal{H}_{k+1}]$$

and $x_{i_C} \in C$ is a representative of class $C \in \mathcal{H}_{k+1}$, while $\hat{X}_{k+1} = \{x_{i_C} : C \in \mathcal{H}_{k+1}\}$. (Note that $H \in \mathcal{H}_k$, while $C \in \mathcal{H}_{k+1}$ and $\hat{X}_{k+1}$ is defined using $\mathcal{H}_{k+1}$.) This implies that R_k is transitive. Indeed, for any $(x_i, x_j), (x_j, x_k) \in R_k$ and $s' \in \mathbb{M}^X$, the previous equation ensures for state $s := s'[x_i \mapsto 0_\oplus, x_k \mapsto s'_{x_i} \oplus s'_{x_k}]$ and any $H \in \mathcal{H}_k$ that

$$\bigoplus_{x_i \in H} f_{x_i}(s) = \bigoplus_{x_i \in H} f_{x_i}(\tilde{s}') = \bigoplus_{x_i \in H} f_{x_i}(\tilde{s}'') = \bigoplus_{x_i \in H} f_{x_i}(\tilde{s}'''),$$

where

$$\tilde{s}' = s[x_l \mapsto 0_\oplus : x_l \notin \hat{X}_{k+1}][x_{i_{C'}} \mapsto \bigoplus_{x_j \in C'} s_{x_j} : C' \in \mathcal{H}_{k+1}],$$
$$\tilde{s}'' = s'[x_l \mapsto 0_\oplus : x_l \notin \hat{X}_{k+1}][x_{i_{C'}} \mapsto \bigoplus_{x_j \in C'} s'_{x_j} : C' \in \mathcal{H}_{k+1}],$$
$$\tilde{s}''' = s'[x_i \mapsto 0_\oplus, x_j \mapsto 0_\oplus, x_k \mapsto s'_{x_i} \oplus s'_{x_j} \oplus s'_{x_k}].$$

Hence, $R^*_k = R_k$ and the expression H/R is indeed well-defined in Algorithm 1. Further, a proof by induction over $k \geq 1$ shows that a) $\mathcal{H}_*$ is a refinement of $\mathcal{H}_k$ and b) $\mathcal{H}_k$ is a refinement of $\mathcal{H}_{k-1}$. Since $\mathcal{H}_*$ is a refinement of any $\mathcal{H}_k$, it holds that $\mathcal{H}_* = \mathcal{H}_k$ if $\mathcal{H}_k$ is a GFB partition. Since X is finite, b) allows us to fix the smallest $k \geq 1$ with $\mathcal{H}_k = \mathcal{H}_{k-1}$. This, in turn, implies that $\mathcal{H}_{k-1}$ is a GFB. To see the complexity statement, we note that the algorithm can perform at most $|X|$ refinements, while each iteration compares $\mathcal{O}(|X|^2)$ pairs. For the decidability, instead, we first note that the finiteness of $\mathbb{M}$ ensures the finiteness of $\oplus \subseteq \mathbb{M} \times \mathbb{M}$ and any $f_{x_i} \subseteq \mathbb{M}^X \times \mathbb{M}$. Hence, checking

$$\bigwedge_{C \in \mathcal{H}} \Big(\bigoplus_{x_k \in C} f_{x_k} = \bigoplus_{x_k \in C} f_{x_k}[x_i/0_\oplus][x_j/(x_i \oplus x_j)] \Big)$$

amounts to a finite number of checks over finite sets and is thus decidable. □

The decidability of $\Psi^{\mathcal{H}}_{x_i,x_j}$ for $\mathbb{M}$ infinite is less immediate. Indeed, since deciding $\Psi^{\mathcal{H}}_{x_i,x_j}$ amounts to deciding identities between functions, decidability over infinite domains critically hinges on the nature of the update functions. For instance, if $\mathbb{M} = \mathbb{R}$, the conditions of $\Psi^{\mathcal{H}}_{x_i,x_j}$ require one to decide the equivalence of real-valued functions. If $\oplus = +$ and update function terms arise through addition and multiplication of variables and may contain minima and maxima expressions, the problem is double exponential [8]. If also exponential and trigonometric functions are allowed, the problem becomes undecidable [38].

We thus study the complexity of deciding $\Psi^{\mathcal{H}}_{x_i,x_j}$ when $(f_{x_i})_{x_i \in X}$ are polynomials and $\oplus \in \{+, \cdot\}$. In such a case, checking $\Psi^{\mathcal{H}}_{x_i,x_j}$ amounts to deciding whether the polynomials

$$\bigoplus_{x_k \in C} f_{x_k} \quad \text{and} \quad \bigoplus_{x_k \in C} f_{x_k}[x_i/0_\oplus][x_j/(x_i \oplus x_j)]$$

are equal. In case of the real and complex field, this question is equivalent to polynomial identity testing for which no holistic algorithms with polynomial time complexity are known [39].[2] Fortunately, the following result readily follows from the Schwartz-Zippel lemma [39].

Theorem 5. *Let $D = (X, F)$ be a discrete-time DS and $\mathcal{X}_R$ a partition. Then, if $(f_{x_i})_{x_i \in X}$ are polynomials over some (sufficiently large) field $\mathbb{M}$ and $\oplus \in \{+, \cdot\}$, Algorithm 1 runs in randomized polynomial time. More specifically, assume that $\Psi^{\mathcal{H}}_{x_i,x_j}$ is false and that it involves polynomials of degree less or equal d. Then, for any finite set $S \subseteq \mathbb{M}$, any $C \in \mathcal{H}$ and a uniformly sampled $v \in S^X$, we have*

$$\mathbb{P}\Big\{ \bigoplus_{x_k \in C} f_{x_k}(v) = \bigoplus_{x_k \in C} f_{x_k}[x_i/0_\oplus][x_j/(x_i \oplus x_j)](v) \Big\} \leq \frac{d}{|S|},$$

where $\mathbb{P}\{A\}$ denotes the probability of event A. In particular, one obtains a polynomial time randomized algorithm whenever $\mathbb{M}$ has more than d elements.

VI. Continuous-time DS

We relate GFB to continuous-time DS, showing how GFB encapsulates existing bisimulations for (nonlinear) ODEs. Thus, in what follows we consider DS with domain $\mathbb{R}$. We can study minimizations for an ODE system $\partial_t v(t) = \Phi(v(t))$ (where ∂_t denotes time derivative) using GFB on its time discretization (X, F), where $F(s) = s + \tau\Phi(s)$. Standard results imply that the approximation error between the ODEs and its time discretization vanishes if τ approaches zero [40].

A. Exact lumpability

GFB-type reductions can be captured by exact lumpability, an established reduction notion for ODEs [23], [24]. Indeed, exact lumping must not be necessarily induced by a partition of the variables. However, we will show that when an exact lumping on an ODE system is described by the homomorphism ψ_R of an equivalence relation R, then it must necessarily be a GFB for its discretization. We start with the definition of exact lumping [23].

Definition 5. *Given an ODE system $\partial_t v(t) = \Phi(v(t))$ with a differentiable function $\Phi : \mathbb{R}^X \to \mathbb{R}^X$, a twice differentiable function $\psi : \mathbb{R}^X \to \mathbb{R}^{\hat{X}}$ is an exact lumping if $|\hat{X}| < |X|$ and there is a unique differentiable function $\hat{\Phi} : \mathbb{R}^{\hat{X}} \to \mathbb{R}^{\hat{X}}$ such*

[2]The common holistic approach rewrites a polynomial into a sum of monomials. Hence, if $\oplus = \cdot$ and all f_{x_k} have, say, 2 monomials, a direct computation of the monomials of $\oplus_{x_k \in C} f_{x_k}$ requires $\mathcal{O}(2^{|C|})$ steps.

that for any $v : [0;T] \to \mathbb{R}^X$ satisfying $\partial_t v(t) = \Phi(v(t))$, it holds that $\partial_t \psi(v(t)) = \hat{\Phi}(\psi(v(t)))$ for all $t \in [0;T]$.

Consider, e.g., the model

$$\partial_t v_{x_1} = v_{x_1}$$
$$\partial_t v_{x_2} = v_{x_2}.$$

Then, $\psi(v_{x_1}, v_{x_2}) = v_{x_1} v_{x_2}$ is an exact lumping since

$$\begin{aligned}\partial_t \psi(v) &= (\partial_{x_1}\psi(v), \partial_{x_2}\psi(v)) \cdot \Phi(v) \\ &= (v_{x_2}, v_{x_1}) \cdot (\partial_t v_{x_1}, \partial_t v_{x_2})^T \\ &= 2v_{x_1} v_{x_2} = 2\psi(v)\end{aligned}$$

where superscript T denotes the transpose of a vector. We can observe that this can be discovered using GFB on the time discretization of the ODE system, given by

$$f_{x_1}(s) = s_{x_1} + \tau s_{x_1} \quad \text{and} \quad f_{x_2}(s) = s_{x_2} + \tau s_{x_2}.$$

Indeed $\mathcal{X}_R = \{\{x_1, x_2\}\}$ is a GFB over $(\mathbb{R}, \cdot)$ since

$$\begin{aligned}f_{x_1} \cdot f_{x_2} &= (x_1 + \tau x_1) \cdot (x_2 + \tau x_2) \\ &= x_1 x_2 + 2\tau x_1 x_2 + \tau^2 x_1 x_2 \\ &= (f_{x_1} \cdot f_{x_2})[x_2/1, x_1/x_1 x_2].\end{aligned}$$

This shows that ψ_R is indeed an exact lumping. The next result formalizes this relationship.

Theorem 6. *Given $\partial_t v(t) = \Phi(v(t))$ with a differentiable function $\Phi : \mathbb{R}^X \to \mathbb{R}^X$, consider the DS $D_\tau = (X, F)$ with $F(s) = s + \tau\Phi(s)$, where $\tau > 0$. Further, let us assume that $\oplus : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is twice differentiable and that $(\mathbb{R}, \oplus)$ is a commutative monoid. Then, for any partition $\mathcal{X}_R$ of X:*

1) *If R is a GFB of all D_τ, then ψ_R is an exact lumpability of $\partial_t v(t) = \Phi(v(t))$.*
2) *If ψ_R is linear, then R is a GFB of all D_τ if and only if ψ_R is an exact lumpability of $\partial_t v(t) = \Phi(v(t))$.*

Proof of Theorem 6. See proof of Theorem 7. □

With the exception of the important case where ψ_R is linear, Theorem 6 does not address whether GFB is also a necessary condition for exact lumpability. Indeed, it turns out that a characterization requires to relax formula $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ to, roughly speaking, ignore the terms of (higher) order $\tau^2, \tau^3, ...$ and so on. This is exemplified next.

Example 7. *Let us fix a continuous DS*

$$\partial_t v_{x_1} = v_{x_1} \log(v_{x_2}) \quad \partial_t v_{x_2} = v_{x_2} \log(v_{x_1}).$$

Together with $v \oplus v' = \log(v) + \log(v')$ and $\mathcal{X}_R = \{\{x_1, x_2\}\}$, it then holds that $\psi_R(v_{x_1}, v_{x_2}) = \log(v_{x_1}) + \log(v_{x_2})$ is an exact lumping, while $\mathcal{X}_R$ is not a GFB.

In order to see this, we start by noting that

$$\psi_R(v_{x_1}, v_{x_2}) = \log(v_{x_1}) + \log(v_{x_2})$$

is an exact lumping because

$$\partial_t \psi_R(v) = (v_{x_1}^{-1}, v_{x_2}^{-1}) \cdot (\partial_t v_{x_1}, \partial_t v_{x_2})^T = \psi_R(v).$$

At the same time, the ODE discretization of the model is

$$f_{x_1} = x_1 + \tau x_1 \log(x_2), \qquad f_{x_2} = x_2 + \tau x_2 \log(x_1).$$

Writing $h = \tau x_1 x_2 \log(x_1 x_2) + \tau^2 x_1 x_2 \log(x_1)\log(x_2)$ for convenience, we observe that

$$\begin{aligned}\log(f_{x_1}) + \log(f_{x_2}) &= \log(f_{x_1} f_{x_2}) \;=\; \log\big(x_1 x_2 + h\big) \\ &= \log(x_1 x_2) + (\partial \log)(x_1 x_2) h + \\ &\qquad (\partial^2 \log)(x_1 x_2) \tfrac{h^2}{2} + \mathcal{O}(\tau^3) \\ &= \log(x_1 x_2) + \tfrac{h}{x_1 x_2} - \tfrac{h^2}{2 x_1^2 x_2^2} + \mathcal{O}(\tau^3) \\ &= \log(x_1 x_2) + \tau \log(x_1 x_2) + \\ &\qquad \tau^2 \log(x_1)\log(x_2) - \tfrac{\tau^2 \log(x_1 x_2)^2}{2} + \mathcal{O}(\tau^3)\end{aligned}$$

Here, $\mathcal{O}$ refers to big O *notation from numerical analysis, while the third identity follows from Taylor's theorem and from $\partial \log(x) = x^{-1}$ and $\partial^2 \log(x) = -x^{-2}$. Since the higher-order term $\tau^2 \log(x_1)\log(x_2)$ cannot be expressed in terms of $\log(x_1 x_2)$, we conclude that $\mathcal{X}_R$ is not a GFB.*

We now characterize exact lumpings of the form ψ_R, accounting for Example 7 and generalizing Theorem 6. As anticipated, we ignore higher-order terms $\mathcal{O}(\tau^2)$ when checking $\Psi^{\mathcal{X}_R}_{x_i,x_j}$, where $\mathcal{O}$ is the *big O* notation from numerical analysis.

Theorem 7. *Given $\partial_t v(t) = \Phi(v(t))$ with a differentiable vector field $\Phi : \mathbb{R}^X \to \mathbb{R}^X$, consider the DS $D_\tau = (X, F)$ with $F(s) = s + \tau\Phi(s)$ where $\tau > 0$. Let us assume that $\oplus : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is twice differentiable and that $(\mathbb{R}, \oplus)$ is a commutative monoid. Then, for any partition $\mathcal{X}_R$ of X, function ψ_R is an exact lumping iff for all $(x_i, x_j) \in R$ with $x_i \neq x_j$ formula $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ is valid up to $\mathcal{O}(\tau^2)$, that is*

$$\begin{aligned}\bigwedge_{C \in \mathcal{X}_R} \Big(\bigoplus_{x_k \in C} f_{x_k} + \mathcal{O}(\tau^2) = \\ \bigoplus_{x_k \in C} f_{x_k}[x_i/0_\oplus][x_j/(x_i \oplus x_j)] + \mathcal{O}(\tau^2)\Big).\end{aligned} \tag{5}$$

Proof of Theorem 7. To improve readability, we write ψ instead of ψ_R in the present proof. Since $\oplus$ is twice differentiable by assumption, so is $\psi = (\psi_H)_{H \in \mathcal{X}_R}$. For any $H \in \mathcal{X}_R$, Taylor's theorem thus ensures

$$\begin{aligned}\psi_H(F(s)) &= \psi_H(s + \tau\Phi(s)) \\ &= \psi_H(s) + (\partial_s \psi_H)(s + \tau\Phi(s)) \cdot \tau\Phi(s) + \mathcal{O}(\tau^2) \\ &= \psi_H(s) + \tau \cdot (\partial_s \psi_H)(s + \tau\Phi(s)) \cdot \Phi(s) + \mathcal{O}(\tau^2)\end{aligned}$$

We begin by assuming that ψ is an exact lumping. Then, with $\partial_t v(t) = \Phi(v(t))$, by [24] the derivative of $t \mapsto \psi_H(v(t))$ can be written as a function of $\big(\psi_C(v(t))\big)_{C \in \mathcal{X}_R}$. Since $v(0) \in \mathbb{R}^X$ can be chosen arbitrarily, there is thus a function $\wp_H$ such that $\wp_H(\psi(s)) = (\partial_s \psi_H)(s + \tau\Phi(s)) \cdot \Phi(s)$ for all $s \in \mathbb{R}^X$. Overall, we conclude for all $s \in \mathbb{R}^X$

$$\psi_H(F(s)) = \psi_H(s) + \tau \cdot \wp_H(\psi(s)) + \mathcal{O}(\tau^2)$$

Since $H \in \mathcal{H}$ can be chosen arbitrarily, following the argumentation from the proof of Theorem 3, we infer that for

all $(x_i, x_j) \in R$ with $x_i \neq x_j$ formula $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ is valid up to $\mathcal{O}(\tau^2)$. For the converse, let us assume that for all $(x_i, x_j) \in R$ with $x_i \neq x_j$ formula $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ is valid up to $\mathcal{O}(\tau^2)$. Then, Taylor's theorem yields as before

$$\psi_H(F(s)) = \psi_H(s) + \tau \cdot (\partial_s \psi_H)(s + \tau\Phi(s)) \cdot \Phi(s) + \mathcal{O}(\tau^2)$$

With this and the validity of the aforementioned $\Psi^{\mathcal{X}_R}_{x_i,x_j}$, the argumentation from the proof of Theorem 3 ensures the existence of functions $(\wp_H)_{H \in \mathcal{X}_R}$ over $\mathbb{R}^{\mathcal{X}_R}$ such that

$$\psi_H(F(s)) = \psi_H(s) + \tau \cdot \wp_H(\psi(s)) + \mathcal{O}(\tau^2)$$

for all $H \in \mathcal{X}_R$ and $s \in \mathbb{R}^X$. Hence, with $\partial_t v(t) = \Phi(v(t))$, the derivative of $t \mapsto \psi_H(v(t))$ can be written as a function of $\big(\psi_C(v(t))\big)_{C \in \mathcal{X}_R}$. Since $v(0) \in \mathbb{R}^X$ can be chosen arbitrarily, we obtain that ψ is an exact lumping. This completes the proof of Theorem 7. We next turn to the proofs of 1) and 2) of Theorem 6. For 1), we note that $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ is valid up to $\mathcal{O}(\tau^2)$ for all $(x_i, x_j) \in R$ when R is a GFB. Instead, for 2) we observe that for a linear ψ_R there are no higher-order terms, i.e., $\mathcal{O}(\tau^2) = 0$. This two observations, combined with the foregoing discussion, yield statements 1) and 2). □

Theorem 6 is related to geometric integration where it has been shown [41, Section IV.1] that discrete-time approximations preserve invariants of continuous-time DS only when these are linear or quadratic, but not if they are cubic or of higher degree. In contrast, Theorem 7 provides a one-to-one correspondence between continuous- and discrete-time invariants by dropping the higher order terms. Additionally, Theorem 6 and 7 allow in contrast to [41] for the algorithmic computation of (nonlinear) invariants.

We end the subsection by noting that if the functions $(f_{x_i})_{x_i \in X}$ are polynomials, then (5) can be checked algorithmically by representing polynomials as sums of monomials and by dropping afterwards all monomials containing a term τ^ν with $\nu \geq 2$. Moreover, we remark that the big-O notation encapsulates in Theorem 7 the universal quantifier across all $\tau > 0$, thus requiring the statements to hold for any positive τ.

B. Forward differential equivalence and Markov chains

Using the results of this section we can relate GFB with analogous bisimulations for DS. We start by restating the notion of forward differential equivalence (FDE) from [8].

Definition 6 (FDE)**.** *Let us consider an ODE system $\partial_t v(t) = \Phi(v(t))$ with a differentiable function $\Phi : \mathbb{R}^X \to \mathbb{R}^X$. A partition $\mathcal{X}_R$ of X is an FDE if ψ_R in case of $\oplus = +$ is an exact lumpability.*

The next result follows from Theorem 6, relating GFB and FDE [8].

Corollary 2. *Given $\partial_t v(t) = \Phi(v(t))$ with a differentiable vector field $\Phi : \mathbb{R}^X \to \mathbb{R}^X$, and the DS $D_\tau = (X, F)$ with $F(s) = s + \tau\Phi(s)$, where $\tau > 0$. Then, for $(\mathbb{R}, +)$, we have that R is a GFB of all D_τ iff R is an FDE of $\partial_t v(t) = \Phi(v(t))$.*

Proof of Corollary 2. Set $\oplus = +$ in Theorem 6. □

Similarly, the next corollary relates GFB with continuous-time Markov chains [4] and probabilistic bisimulation of discrete-time Markov chains [3].

Corollary 3. *Let (X, Q) be a continuous-time Markov chain with states X and transition rate matrix $Q \in \mathbb{R}^{X \times X}$. Consider the DS $D_\tau = (X, F)$ with $F(s) = s + \tau Q^T s$ where $\tau > 0$. Then, D_τ is an embedded discrete-time Markov chain of (X, Q) for sufficiently small $\tau > 0$. With this, for monoid $(\mathbb{R}, +)$ the following three conditions are equivalent: 1. R is a GFB of all D_τ; 2. R is an ordinary lumpability of (X, Q); 3. R is a probabilistic bisimulation of all D_τ that describe a discrete-time Markov chain.*

Proof of Corollary 3. The vector of transient probabilities of the Markov chain at time $t \geq 0$ satisfies the forward Kolmogorov equations $\partial_t \pi(t) = Q^T \pi(t)$. Moreover, by [8], an equivalence relation R over X is an ordinary lumpability if and only if R is an FDE the forward Kolmogorov equations. With this, Corollary 2 yields the equivalence of 1) and 2). The equivalence of 2) and 3), instead, is a well-known fact [4]. □

Remark 2. *The above discussion ensures that $\Psi^{\mathcal{H}}_{x_i,x_j}$ from Algorithm 1 can be decided in polynomial time for FDE and probabilistic bisimulation, see [8].*

C. Attractors of continuous-time DS

The notion of attractor from Definition 2 also exists for continuous-time dynamics [42].

Definition 7 (Attractor)**.** *Consider an ODE system $\partial_t v(t) = \Phi(v(t))$ with a differentiable vector field $\Phi : \mathbb{R}^X \to \mathbb{R}^X$. A compact nonempty set $A \subseteq \mathbb{R}^X$ is an attractor (aka asymptotically stable) if there exists an open neighborhood B of A such that for any $\varepsilon > 0$ there is some time $t' \geq 0$ such that for any $v[0] \in B$, the solution of $\partial_t v(t) = \Phi(v(t))$ with $v(0) = v[0]$ satisfies $d(v(t), A) \leq \varepsilon$ for all $t \geq t'$. Here, $d(v(t), A) = \min_{a \in A} d(v(t), a)$ and distance d is induced, similarly to B, by some norm.*

The next result from [42] essentially ensures that attractors of an ODE system can be approximated by attractors of its discrete-time discretization.

Theorem 8 ([42])**.** *Given $\partial_t v(t) = \Phi(v(t))$ with a differentiable vector field $\Phi : \mathbb{R}^X \to \mathbb{R}^X$, let $A \subseteq \mathbb{R}^X$ be an attractor of $\partial_t v(t) = \Phi(v(t))$. Then, for any $\tau > 0$, there exists a set $A(\tau) \subseteq \mathbb{R}^X$ such that*

- *$F(A(\tau)) \subseteq A(\tau)$, where $F(s) = s + \tau\Phi(s)$ and;*
- *The sets $A(\tau)$ converge to the set A in the Hausdorff metric as $\tau \to 0$.*

Corollary 1 and Theorem 8 allow to use GFB to argue on attractors of ODEs. Less importantly, Theorem 8 does not explicitly provide basins of attraction for the sets $A(\tau)$. However, $A(\tau)$ are attractors when the discrete topology is used.

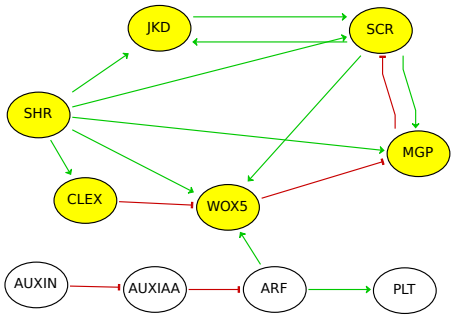


Fig. 1. Pictorial representation of the Boolean network from Example 1 using GinSim [43], adapted from [37].

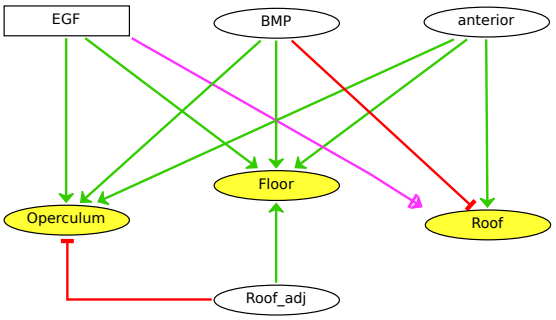


Fig. 2. Pictorial representation using GinSim [43] adapted from [44] of the model on eggshell formation for drosophila melanogaster flies.

VII. Applications

A. Regulatory Networks

We now apply GFB to Boolean and multi-valued networks from the literature.

a) ***BN case study****:* We study the BN used as running example (Example 1). To ease interpretation, Fig. 1 uses the typical graphical notation of *influence graphs* (offered, e.g., by GinSIM [43]). Nodes denote variables, while arrows denote *influences* among nodes. Influences come from the update functions: green and red arrows denote, respectively, positive (*promotion*) and negative (*inhibition*) influence. In Example 1, ARF promotes PLT due to term ARF in f_{PLT}, while AUXIN inhibits AUXIAA due to term $\neg$AUXIN in f_{AUXIAA}. The BN consists of two connected pathways: one for the transcription factor SHR with its signalling to the other variables of the pathway (we highlight in yellow the involved nodes), and one involving the hormone AUXIN and its signaling to the plethora (PLT) genes.

BN variables can be categorized into three groups [45]: *inputs* (SHR, and AUXIN) that do not have incoming edges, *outputs* (PLT) that do not have outgoing edges, and the remaining *internal nodes*. The distinction is obvious from update functions: inputs have constant update functions, while outputs do not appear in the update function of other variables. Inputs are often set by the modeler to perform *what if* experiments, whereas outputs permit to observe the response dynamics of the model. In this BN, each input *controls* its own pathway, meaning that the modeller can decide to enable them via appropriate initial states.

Considering the GFB $\mathcal{X}_R$ from Example 3 for $\oplus = \wedge$, the only non-trivial block $C = \{\mathsf{SCR}, \mathsf{SHR}, \mathsf{JKD}, \mathsf{MGP}, \mathsf{WOX5}, \mathsf{CLEX}\}$ corresponds to the yellow nodes in Fig. 1. This GFB is computed using the initial partition with two blocks separating outputs and non-output nodes. Considering the reduced model for $\mathcal{X}_R$ from Example 6, all yellow nodes in Fig. 1 get collapsed into one, meaning that the SHR pathway is abstracted away. In other words, in this example GFB has automatically identified and *simplified* a pathway in the model, offering a coarser representation of the system focusing on the AUXIN pathway only.

b) ***Multi-valued network case study****:* We apply GFB to a multi-valued regulatory network (MV) from [44]. Intuitively, an MV is a BN where variables can admit more than two values. This is a single-cell model describing the development of eggshell structures in drosophila melanogaster flies. The MV has 7 variables with relations depicted in Fig. 2 and update functions:

$$
\begin{aligned}
f_{\mathsf{EGF}} &= \mathsf{EGF} \\
f_{\mathsf{BMP}} &= \mathsf{BMP} \\
f_{\mathsf{Ant}} &= \mathsf{Ant} \\
f_{\mathsf{RoofAdj}} &= \mathsf{RoofAdj} \\
f_{\mathsf{Roof}} &= \mathsf{Ant}:1 \wedge \mathsf{EGF}:1 \wedge \mathsf{BMP}:0 \\
f_{\mathsf{Floor}} &= \mathsf{Ant}:1 \wedge (\mathsf{EGF}:2 \vee (\mathsf{EGF}:1 \wedge \mathsf{BMP}:1)) \wedge \mathsf{RoofAdj}:1 \\
f_{\mathsf{Operc}} &= \mathsf{Ant}:1 \wedge (\mathsf{EGF}:2 \vee (\mathsf{EGF}:1 \wedge \mathsf{BMP}:1)) \wedge \mathsf{RoofAdj}:0
\end{aligned}
$$

Using the notation in [44], "$\mathsf{var} : v$" stands for *variable* var *has value* v. This is a Boolean predicate evaluating to 1 if var has value v, and 0 otherwise. Variable EGF, the rectangular node in Fig. 2, can take values 0, 1, 2, denoting absent/intermediate/high activation levels. All other variables are Boolean (0/1).[3]

Differently from Fig. 1, variables divide in two groups only: the *inputs* EGF, BMP, Ant, and RoofAdj, and the *outputs* Operc, Floor, and Roof. We also have a third edge type, the purple one from EGF to Roof. This visually stresses that EGF influences Roof only when in intermediate level and not when in high level.

The MV relates three follicle cell fates, the outputs, to combinations of values of the inputs. EGF and BMP are known pathways responsible for patterning of the drosophila eggshell [44]. This is encoded in the model because EGF and BMP influence, in different ways, all outputs. Finally, Ant models the anterior competence region, therefore it is required by all outputs, while RoofAdj accounts for the state

[3]Our framework requires all variables to have same domain $\mathbb{M}$. In order to support MV, we implicitly *expand* the domain of all variables to the largest one (e.g., $\{0, 1, 2\}$ of EGF). This does not change the models' dynamics, in the sense that when setting initial states fitting in the original domain we will remain within the original domain.

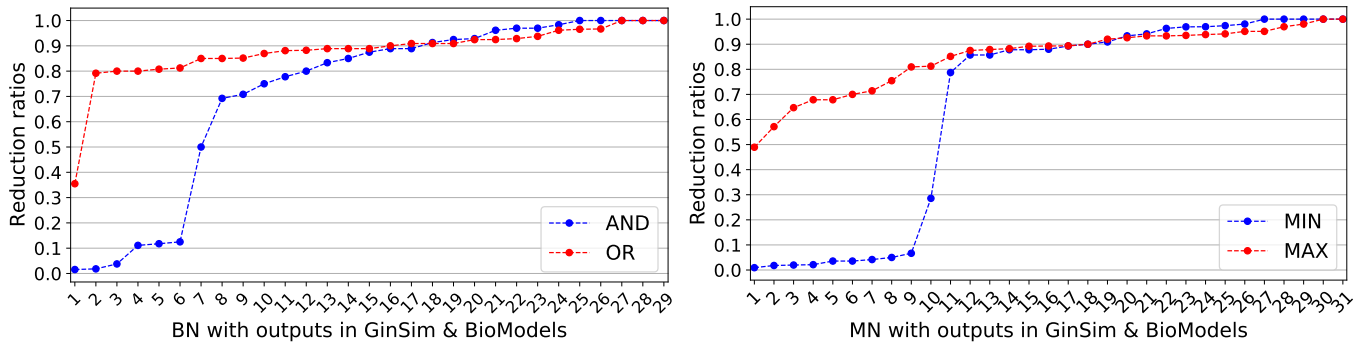


Fig. 3. **(Left)** Reduction ratios (reduced variables over original ones) in ascending order for the 29 BN with outputs from GINsim and BioModelsDB for $\oplus \in \{\wedge, \vee\}$ and initial partitions with two blocks separating output and non-outputs. **(Right)** Same as (Left) for the 31 MV with outputs from the two repositories using $\oplus \in \{\min, \max\}$.

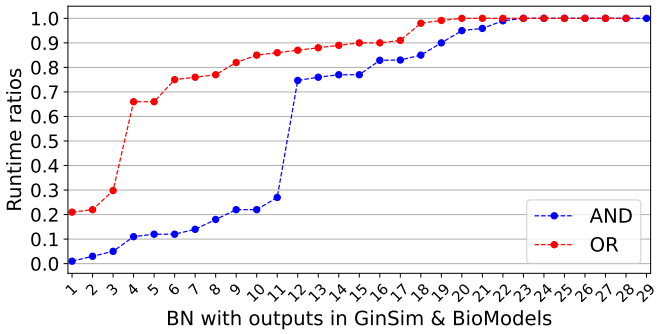


Fig. 4. Runtime ratios in ascending order for computation of attractors for the 29 BNs from Fig. 3 (Left) and their reductions.

of neighboring cells by promoting Floor and inhibiting Operc (*operculum*).

The partition with one block for all outputs and singleton blocks for each input is a GFB for $\oplus \in \{\max, \min\}$. By Definition 4, we get two different reduced models in the two cases, enabling complementary studies. Case $\max$ allows *full output deactivation* studies, meaning that the reduced variable for the outputs gets value 0 only if all outputs have value 0. Instead, case $\min$ allows *full output activation* studies, as the reduced variable gets value 1 only when all outputs have value 1. By naming outputs the reduced variable corresponding to the block of outputs, by applying Definition 4 and some algebraic simplification we get:

$$f_{\text{outputs}} = \text{Ant}{:}1 \wedge (\text{EGF}{:}1 \vee \text{EGF}{:}2), \qquad \text{for } \oplus = \max,$$
$$f_{\text{outputs}} = 0, \qquad \text{for } \oplus = \min,$$

while the update functions of the input variables remain unchanged. From this we get that: despite the three outputs have different dependencies on Ant, BMP, RoofAdj, and on different values of EGF, in the $\oplus = \max$ case it is enough to consider only ANT and EGF to answer questions related to full output deactivation. Furthermore, it is not necessary anymore to use three values for EGF, as we are only interested in the cases in which it is 0 or positive ($\text{EGF}:1 \vee \text{EGF}:2$). Instead, from the $\oplus = \min$ case we know that the original model never expresses cases of full activation, i.e., it never happens that the three outputs have all value 1. Indeed, by studying the update functions of the original outputs, we see that there are no values for the involved variables that makes all of them true.

*c) **Large-scale validation of GFB on regulatory networks**:* We present a large-scale validation of GFB on the BNs and MVs from the repositories GinSim (ginsim.org/models_repository) and BioModelsDB [20]. We validate GFB in terms of aggregation power and of speed-up offered for attractor analysis.

Experimental setting. We created a prototype implementation of GFB integrated with the SMT solver Z3 [46] to check formulas from $\Psi^{\mathcal{H}}_{x_i,x_j}$ in Algorithm 1.[4] We created an importer for SBML Qual [47], an XML format supported by both repositories, allowing us to import all 43 BNs and 50 MVs. In order to obtain physically-relevant initial partitions, we infer *candidate outputs*, variables not appearing in the update function of other variables. For each model, we create *output-preserving* initial partitions: these consist of two blocks, one containing all outputs and one containing the remaining variables. This guarantees that reduced models allow, e.g., for full output (de)activation studies discussed before. In order to perform a consistent treatment, we restricted our analysis on the 29 BNs and 31 MVs with at least one candidate output.

Validation of aggregation power. Fig. 3 (Left) provides the reduction ratios obtained for the BNs using $\oplus \in \{\wedge, \vee\}$. For each model we plot the reduction ratio, defined as the number of reduced variables over that of original ones. For each operator $\oplus$, the ratios were sorted in ascending order. We can see that $\oplus = \wedge$ has high aggregation power, with about one third of the models having reduction ratio below 0.6, while for $\oplus = \vee$ most of the models have 0.8 or more. For $\oplus = \wedge$, some models have particularly low ratios, below 0.2, some of which due to the fact that the reduced model has 2 variables only. We remark that these shall not be considered *degenerate* reductions, because of the used initial partitions, as discussed. We do not present results on maximal reductions,

[4]Note to reviewers: the tool, models, and replication material have been omitted to adhere to double-blind policy. They will be made available online upon acceptance.

obtained with the initial partition with one block only. These are significantly smaller, but some are degenerate with one variable only. We leave for future work a detailed study on finer intermediate reductions using model-specific initial partitions preserving variables of interest for the modeler. For example, a modeler could be interested in preserving only some outputs. Fig. 3 (Right) presents a similar study performed on the MVs using $\oplus = \min$ and $\oplus = \max$, confirming the aggregation power of GFB.

Validation of analysis speed-up. Corollary 1 ensures that GFB maps all attractors of the original system to attractors of the reduced one. Here we show that this can speed-up attractor computation. We use the COLOMOTO Notebook [48], an environment incorporating a variety of tools for BN analysis. An example is BNS [49], which combines SAT-solving and bounded model checking to identify attractors. We computed the attractors of the 29 considered BNs and of their reductions. We could not consider MVs because we are not aware of tools for general attractor analysis for MVs. Fig. 4 shows the obtained runtime ratios (computation time of attractors in the reduced model over that in the original one). In several cases the reduction led to significant analysis speed-ups: in 11 BNs the ratio is less than 0.3. We remark that GFB is *useful*, because the analysis of the original BNs, the AND- and OR-reductions took on average 100s, 30s and 60s, respectively. Notably, reductions with low reduction ratios are particularly fast (fewer algorithm iterations): the 6 AND-reductions in Fig. 3 (Left) with ratio smaller than 0.3 take less than 1.5 seconds on average.

*d) **Enabling analysis of large BNs using GFB**:* We now apply GFB to a large BN of signalling pathways central to macrophage activation [50]. This BN contains *321* variables, making attractor computation infeasible even using the most efficient tool for this task [49]. In particular, the analysis does not terminate within an arbitrarily chosen time limit of 10 hours. Our crucial hypothesis is that *GFB can enable some analysis* of this otherwise not analyzable BN, although with certain restrictions imposed by what is exactly preserved by the reduction.

The results are presented in Table I. In this experiment we focus on $\oplus = \wedge$. We can see that the maximal reduction is not physically-relevant, as it reduces to 1 variable only. The output-preserving reduction, instead, leads to a reduced model with 189 variables. Despite this, the obtained reduced model is still not analyzable within the chosen time limit. We now show how two alternative initial partitions lead to reduced models that can be effectively analyzed. In particular, we assume that the modeler is not interested in preserving all 68 outputs, but two different subsets of them: $O_1 = \{\texttt{S\_28}, \texttt{S\_26}, \texttt{S\_198}, \texttt{S\_11}\}$ and $O_2 = \{\texttt{S\_184}, \texttt{S\_188}\}$. In both cases, we use an initial partition with one block for the selected outputs, and one for all the other variables. In these two cases, we obtained models with 70 and 33 variables, respectively, which admit analysis. In particular, the obtained reduced models can now be analyzed using less than a second.

Model	*Variables*	*Attractors analysis*	
		Count	*Runtime(s)*
Original	321	*—Time Out—*	
Output separated	189	*—Time Out—*	
O1	70	64	0.668
O2	33	64	0.325
Maximal	1	1	0.001

TABLE I
GFB ENABLES ATTRACTORS COMPUTATION ON LARGE BN [50].

B. Non-linear reductions of Differential and Difference Equations

We present examples of exact lumping where ψ_R is not linear, and thus cannot be captured by linear lumpings such FDE. We use $(\mathbb{R}, \cdot)$ with neutral element 1.

*a) **Nonlinear Reduction of a Lotka-Volterra Model over** $(\mathbb{R}, \cdot)$:* We start considering a prototypical higher-order Lotka-Volterra model [21] where x_1 preys x_2 and x_3, while x_2 and x_3 prey together x_1. The corresponding ODE system is

$$\begin{aligned} \partial_t v_{x_1} &= v_{x_1}(1 - v_{x_2} v_{x_3}), \\ \partial_t v_{x_2} &= v_{x_2}(1 - v_{x_1}), \\ \partial_t v_{x_3} &= v_{x_3}(1 - v_{x_1}). \end{aligned} \tag{6}$$

The ODE discretization of (6) is given by

$$\begin{aligned} f_{x_1}(s) &= s_{x_1} + \tau s_{x_1}(1 - s_{x_2} s_{x_3}), \\ f_{x_2}(s) &= s_{x_2} + \tau s_{x_2}(1 - s_{x_1}), \\ f_{x_3}(s) &= s_{x_3} + \tau s_{x_3}(1 - s_{x_1}). \end{aligned}$$

By Theorem 6, the *nonlinear* function $\psi_R(v_{x_1}, v_{x_2}, v_{x_3}) = (v_{x_1}, v_{x_2} \cdot v_{x_3})$ is an exact lumping of (6). Indeed, $\mathcal{X}_R = \{\{x_1\}, \{x_2, x_3\}\}$ is a GFB of (6) for $\oplus = \cdot$ because $\Psi^{\mathcal{X}_R}_{x_2,x_3}$ is valid thanks to the identities $f_{x_1} = f_{x_1}[x_2/1, x_3/x_2 x_3]$, and

$$\begin{aligned} f_{x_2} \cdot f_{x_3} &= (x_2 + \tau x_2(1 - x_1)) \cdot (x_3 + \tau x_3(1 - x_1)) \\ &= x_2 x_3 + 2\tau x_2 x_3(1 - x_1) + \tau^2 x_2 x_3 (1 - x_1)^2 \\ &= (f_{x_2} \cdot f_{x_3})[x_2/1, x_3/x_2 x_3]. \end{aligned}$$

The lumped ODE system is given by $\partial_t v_{x_1} = v_{x_1}(1 - v_{x_2} v_{x_3})$ and

$$\begin{aligned} \partial_t(v_{x_2} v_{x_3}) &= \partial_t v_{x_2} \cdot v_{x_3} + v_{x_2} \cdot \partial_t v_{x_3} \\ &= v_{x_2}(1 - v_{x_1}) v_{x_3} + v_{x_2} v_{x_3}(1 - v_{x_1}) \\ &= 2 v_{x_1} v_{x_2}(1 - v_{x_1}). \end{aligned}$$

*b) **Nonlinear Reduction of Dynamical Weighted Networks over** $(\mathbb{R}, \cdot)$:* We now consider real-valued DS obtained from weighted networks from the Netzschleuder repository [22]. We considered all 72 weighted networks with at most 200 nodes (by restricting to at most the first 15 models from each family of models). The undirected ones were expanded in directed by replacing every undirected edge with two corresponding directed ones with same weight.

We consider two different dynamical interpretations. For A the adjacency matrix of a network, we study the discrete-time DS $x(t+1) = Ax(t)$, and the (ODE discretization

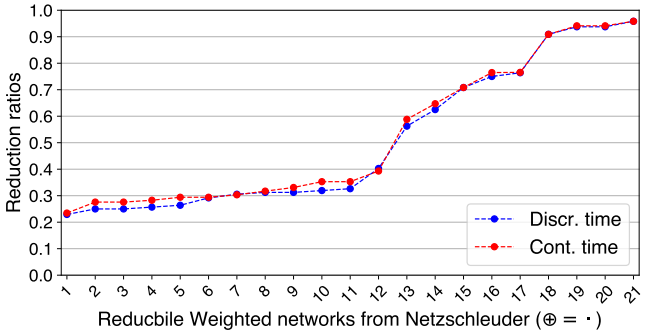


Fig. 5. Similarly to Fig. 3, we plot reduction ratios for the discrete- and continuous-time dynamical interpretations of 21 weighted networks from Netzschleuder. We used one operator, $\oplus = \cdot$, and initial partitions separating the first node in the network from the others.

of the) continuous-time DS $\partial_t v(t) = Av(t)$. In both cases, we use one variable per node.[5] We use $\oplus = \cdot$, obtaining nonlinear reductions. This leads to high nonlinearities in formulas $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ from Theorem 3, complex to handle for Z3. Indeed, Algorithm 1 failed to terminate within an arbitrarily chosen time-out of 1 hour even for models of moderate size. Hence, for our experiments we used the randomized version of the algorithm discussed in Section V, performing 40 tests per formula $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ from Theorem 3 after sampling values for all variables. Currently, our prototype is still based on Z3, to which we provide the sampled values making all $\Psi^{\mathcal{X}_R}_{x_i,x_j}$ formulas variable free. In this setting, Z3 never failed.

Fig. 5 provides the results for the 21 networks that admitted a reduction, 29% of the 72 considered. We got similar reduction ratios in the two interpretations, with slightly better ones for the discrete-time one. The lower reduction power of the continuous-time case comes from two factors: (i) Models have higher nonlinearities due to the τ term; (ii) Theorem 6 gives only a necessary condition for aggregation in this case (our prototype does not support the results of Theorem 7). The largest runtimes for the continuous- and discrete-time cases were about 500 and 400 seconds, respectively, for a model with 145 nodes.

VIII. Conclusion

Generalized forward bisimulation (GFB) is a technique for dimensionality reduction of discrete- and continuous-time dynamical systems that captures and generalizes existing techniques. GFB allows to compute nonlinear reductions. One needs to specify a dynamical system, a commutative monoid (the variables' domain and an operation used to aggregate them), and an initial partition of the variables (used to tune the reduction power to preserve variables of interest). A partition refinement algorithm then minimizes the system over the operation of the monoid. We implemented GFB and applied it to four popular formalisms: difference and differential equations with monoid $(\mathbb{R}, \cdot)$, Boolean networks with $(\mathbb{B}, \wedge)$ and $(\mathbb{B}, \vee)$, multi-valued networks with $(\{0,1,2\}, \min)$ and $(\{0,1,2\}, \max)$. In all cases, GFB yielded notable nonlinear reductions. On 60 Boolean and multi-valued networks from two popular repositories, we have shown high aggregation power and analysis speed-ups. Using an existing large Boolean network with 321 variables we have shown that GFB might enable the analysis of otherwise untractable models. On 21 ODEs originated from weighted networks from a popular repository, we have computed nonlinear reductions thanks to the $\cdot$ operation, showing high aggregation power.

[5] In the continuous-time case, we also have an additional variable for the τ term from the ODE discretization, to which we give constant update function. This guarantees that the obtained reductions hold for any value of τ.

References

[1] D. Sangiorgi, *Introduction to Bisimulation and Coinduction*. Cambridge University Press, 2011.

[2] D. Park, "Concurrency and automata on infinite sequences," in *Theoretical Computer Science*, 1981, pp. 167–183.

[3] K. G. Larsen and A. Skou, "Bisimulation through probabilistic testing," *Inf. Comput.*, vol. 94, no. 1, pp. 1–28, 1991.

[4] P. Buchholz, "Exact and ordinary lumpability in finite Markov chains," *Journal of Applied Probability*, vol. 31, no. 1, pp. 59–75, 1994.

[5] R. De Nicola, U. Montanari, and F. Vaandrager, "Back and forth bisimulations," in *CONCUR '90 Theories of Concurrency: Unification and Extension*, ser. Lecture Notes in Computer Science, J. Baeten and J. Klop, Eds. Springer Berlin Heidelberg, 1990, vol. 458, pp. 152–165. [Online]. Available: http://dx.doi.org/10.1007/BFb0039058

[6] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin, "Forward and backward bisimulations for chemical reaction networks," in *26th International Conference on Concurrency Theory, CONCUR*, 2015, pp. 226–239. [Online]. Available: http://cse.lab.imtlucca.it/~mirco.tribastone/papers/concur2015.pdf

[7] L. Cardelli, I. C. Pérez-Verona, M. Tribastone, M. Tschaikowski, A. Vandin, and T. Waizmann, "Exact maximal reduction of stochastic reaction networks by species lumping," *Bioinform.*, vol. 37, no. 15, pp. 2175–2182, 2021.

[8] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin, "Symbolic computation of differential equivalences," in *POPL*, 2016, pp. 137–150. [Online]. Available: https://doi.org/10.1145/2837614.2837649

[9] R. Paige and R. E. Tarjan, "Three partition refinement algorithms," *SIAM Journal on Computing*, vol. 16, no. 6, pp. 973–989, 1987.

[10] M. S. Okino and M. L. Mavrovouniotis, "Simplification of mathematical models of chemical reaction systems," *Chemical Reviews*, vol. 2, no. 98, pp. 391–408, 1998.

[11] A. Antoulas, *Approximation of Large-Scale Dynamical Systems*, ser. Advances in Design and Control. SIAM, 2005.

[12] T. J. Snowden, P. H. van der Graaf, and M. J. Tindall, "Methods of model reduction for large-scale biological systems: A survey of current methods and trends," *Bulletin of Mathematical Biology*, vol. 79, no. 7, pp. 1449–1486, 2017. [Online]. Available: https://doi.org/10.1007/s11538-017-0277-2

[13] G. Li, H. Rabitz, and J. Tóth, "A general analysis of exact nonlinear lumping in chemical kinetics," *Chemical Engineering Science*, vol. 49, no. 3, pp. 343–361, 1994. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0009250994870063

[14] S. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," *Journal of Theoretical Biology*, vol. 22, no. 3, pp. 437 – 467, 1969.

[15] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin, "Maximal aggregation of polynomial dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 114, no. 38, pp. 10 029 – 10 034, 2017.

[16] M. Hopfensitz, C. Müssel, M. Maucher, and H. A. Kestler, "Attractors in boolean networks: a tutorial," *Computational Statistics*, vol. 28, no. 1, pp. 19–36, 2013.

[17] R. Thomas, D. Thieffry, and M. Kaufman, "Dynamical behaviour of biological regulatory networks — i. biological role of feedback loops and practical use of the concept of the loop-characteristic state," *Bulletin of mathematical biology*, vol. 57, no. 2, pp. 247–276, 1995.

[18] G. Argyris, A. Lluch Lafuente, M. Tribastone, M. Tschaikowski, and A. Vandin, "Reducing boolean networks with backward boolean equivalence," in *International Conference on Computational Methods in Systems Biology*. Springer, 2021, pp. 1–18.

[19] A. Naldi, D. Berenguier, A. Fauré, F. Lopez, D. Thieffry, and C. Chaouiya, "Logical modelling of regulatory networks with ginsim 2.3," *Biosystems*, vol. 97, no. 2, pp. 134–139, 2009.

[20] R. S. Malik-Sheriff, M. Glont, T. V. N. Nguyen, K. Tiwari, M. G. Roberts, A. Xavier, M. T. Vu, J. Men, M. Maire, S. Kananathan, E. L. Fairbanks, J. P. Meyer, C. Arankalle, T. M. Varusai, V. Knight-Schrijver, L. Li, C. Dueñas-Roca, G. Dass, S. M. Keating, Y. M. Park, N. Buso, N. Rodriguez, M. Hucka, and H. Hermjakob, "BioModels — 15 years of sharing computational models in life science," *Nucleic Acids Research*, vol. 48, no. D1, pp. D407–D415, 1 2020, gkz1055. [Online]. Available: https://doi.org/10.1093/nar/gkz1055

[21] P. Singh and G. Baruah, "Higher order interactions and species coexistence," *Theoretical Ecology*, vol. 14, pp. 71–83, 2021.

[22] T. P. Peixoto, "The netzschleuder network catalogue and repository," 2020. [Online]. Available: https://networks.skewed.de/

[23] G. Li and H. Rabitz, "A general analysis of exact lumping in chemical kinetics," *Chemical Engineering Science*, vol. 44, no. 6, pp. 1413–1430, 1989.

[24] A. S. Tomlin, G. Li, H. Rabitz, and J. Tóth, "The effect of lumping and expanding on kinetic differential equations," *SIAM Journal on Applied Mathematics*, vol. 57, no. 6, pp. 1531–1556, 1997.

[25] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin, "Guaranteed error bounds on approximate model abstractions through reachability analysis," in *QEST*, 2018, pp. 104–121.

[26] K. Ghorbal and A. Platzer, "Characterizing algebraic invariants by differential radical invariants," in *TACAS*, E. Ábrahám and K. Havelund, Eds., vol. 8413. Springer, 2014, pp. 279–294.

[27] E. Bartocci, L. Kovács, and M. Stankovic, "Automatic generation of moment-based invariants for prob-solvable loops," in *ATVA*, Y. Chen, C. Cheng, and J. Esparza, Eds., 2019, pp. 255–276.

[28] M. Boreale, "Algebra, coalgebra, and minimization in polynomial differential equations," *Log. Methods Comput. Sci.*, vol. 15, no. 1, 2019.

[29] ——, "Complete algorithms for algebraic strongest postconditions and weakest preconditions in polynomial odes," *Sci. Comput. Program.*, vol. 193, p. 102441, 2020.

[30] G. Bacci, G. Bacci, K. G. Larsen, M. Tribastone, M. Tschaikowski, and A. Vandin, "Efficient local computation of differential bisimulations via coupling and up-to methods," in *Symposium on Logic in Computer Science, LICS*, 2021, pp. 1–14.

[31] G. J. Pappas, G. Lafferriere, and S. Sastry, "Hierarchically consistent control systems," *IEEE Trans. Automat. Contr.*, vol. 45, no. 6, pp. 1144–1160, 2000.

[32] G. J. Pappas and S. Simic, "Consistent abstractions of affine control systems," *IEEE Trans. Automat. Contr.*, vol. 47, no. 5, pp. 745–756, 2002.

[33] A. J. van der Schaft, "Equivalence of dynamical systems by bisimulation," *IEEE Transactions on Automatic Control*, vol. 49, 2004.

[34] A. Naldi, E. Remy, D. Thieffry, and C. Chaouiya, "Dynamically consistent reduction of logical regulatory graphs," *Theoretical Computer Science*, vol. 412, no. 21, pp. 2207–2218, 2011.

[35] A. Veliz-Cuba, "Reduction of boolean network models," *Journal of theoretical biology*, vol. 289, pp. 167–172, 2011.

[36] J. Milnor, "On the concept of attractor," *Communications in Mathematical Physics*, vol. 99, no. 2, pp. 177–195, 1985.

[37] E. Azpeitia, M. Benítez, I. Vega, C. Villarreal, and E. R. Alvarez-Buylla, "Single-cell and coupled grn models of cell patterning in the arabidopsis thaliana root stem cell niche," *BMC systems biology*, vol. 4, no. 1, pp. 1–19, 2010.

[38] D. Richardson, "Some undecidable problems involving elementary functions of a real variable," *The Journal of Symbolic Logic*, vol. 33, no. 4, pp. 514–520, 1968.

[39] N. Saxena, "Progress on polynomial identity testing," *Bull. EATCS*, vol. 99, pp. 49–79, 2009.

[40] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1971.

[41] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration*, 2006.

[42] P. E. Kloeden and J. Lorenz, "Stable attracting sets in dynamical systems and in their one-step discretizations," *SIAM Journal on Numerical Analysis*, vol. 23, no. 5, pp. 986–995, 1986.

[43] A. Naldi, D. Berenguier, A. Fauré, F. Lopez, D. Thieffry, and C. Chaouiya, "Logical modelling of regulatory networks with ginsim 2.3," *Biosystems*, vol. 97, no. 2, pp. 134–139, 2009.

[44] A. Fauré, B. Vreede, E. Sucena, and C. Chaouiya, "A discrete model of drosophila eggshell patterning reveals cell-autonomous and juxtacrine effects," *PLoS Comput Biol*, vol. 10, p. e1003527, 2014.

[45] A. Naldi, P. T. Monteiro, and C. Chaouiya, "Efficient handling of large signalling-regulatory networks by focusing on their core control," in *International Conference on Computational Methods in Systems Biology*. Springer, 2012, pp. 288–306.

[46] L. De Moura and N. Bjørner, "Z3: An efficient smt solver," in *International conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2008, pp. 337–340.

[47] C. Chaouiya, D. Bérenguier, S. M. Keating, A. Naldi, M. P. Van Iersel, N. Rodriguez, A. Dräger, F. Büchel, T. Cokelaer, B. Kowal *et al.*, "SBML qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools," *BMC systems biology*, vol. 7, no. 1, pp. 1–15, 2013.

[48] A. Naldi, P. T. Monteiro, C. Müssel, C. for Logical Models, Tools, H. A. Kestler, D. Thieffry, I. Xenarios, J. Saez-Rodriguez, T. Helikar, and C. Chaouiya, "Cooperative development of logical modelling standards and tools with colomoto," *Bioinformatics*, vol. 31, no. 7, pp. 1154–1159, 2015.

[49] E. Dubrova and M. Teslenko, "A sat-based algorithm for finding attractors in synchronous boolean networks," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 8, no. 5, pp. 1393–1399, 2011.

[50] S. Raza, K. A. Robertson, P. A. Lacaze, D. Page, A. J. Enright, P. Ghazal, and T. C. Freeman, "A logic-based diagram of signalling pathways central to macrophage activation," *BMC systems biology*, vol. 2, no. 1, pp. 1–15, 2008.

C.1 Supplementary Material

The state space and the dynamics of a BN are encoded into the state transition graph (STG) which consists of states and transitions. Consider the introductory example of C given in the top-left part of Fig. 7. The corresponding STG is provided in the bottom left part of figure. Each box contains an evaluation of the set of variables while the arrows correspond to the transitions after an update. For example, if the variables at time point k have the value $(1, 0, 0)$ i.e. $x_1(k) = 1$, $x_2(k) = 0$, and $x_3(k) = 0$-top left blue box-, then by applying the update functions we arrive at the yellow state $(0, 1, 1)$.

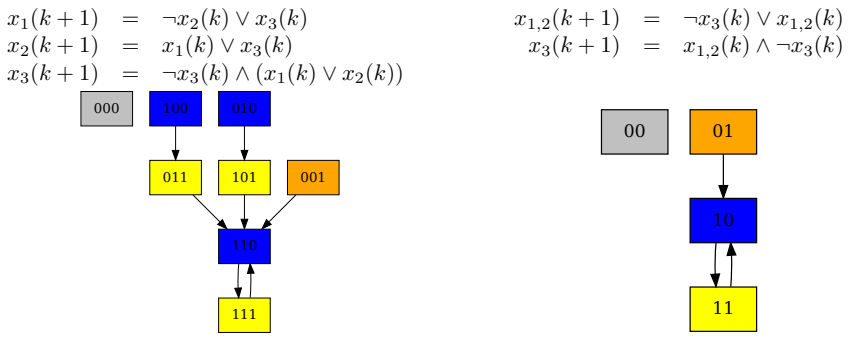


Figure 7: A BN (top-left), its STG (bottom-left), the GFB-reduced BN (top-right) and its (reduced) STG (bottom-right).

We provide the GFB-reduced BN in the top-right part of Fig. 7, as derived in the introduction. Essentially, each state/box of the reduced STG results from original STG after collapsing the first and the second digit of each box to one single digit according to their OR. States with the same colour in the original STG are mapped to the same state in the reduced STG. We highlight that the original and the reduced STG are deterministic and bisimulation equivalent which means that reachability properties are preserved.

Hypothesis. Note that STGs suffer from the state space explosion; there are exponentially many states on the number of the BN variables. In several cases, STG generation takes a lot of time, while in other cases the STG can not be generated due to our limited computational resources. Our crucial hypothesis is that our reduction method can speed-up STG generation and render it feasible.

Configuration. To generate the STG, we use PyBoolNet [29], a python package for the generation, analysis and visualisation of Boolean networks. We focus on *5* BNs of intermediate size wherein the results were outstanding and compute the time needed for the generation of the original and the reduced STGs. The experiments have been conducted in a common PC with an Intel Xeon(R) 2.80GHz and 32GB of RAM. We note that STG generation was infeasible for BNs with more than *24* variables.

Results. We present the results of our analysis in Table 2. The first column (*Model*) contains a model identifier and the relevant literature. The column *original model* contains the number of variables (*Size*) of the original model and the time needed for the generation of the STG in seconds. The same information are presented for the case of the And-Reduced model and the Or-Reduced model in columns 2 and 3.

Model	*Original model*		*And-Reduced*		*Or-Reduced*	
	Size	*STG generation(s)*	*Size*	*STG generation(s)*	*Size*	*STG generation(s)*
M1 [30]	24	*out of memory*	17	9	24	*out of memory*
M2 [30]	26	*out of memory*	18	36	21	188
M3 [30]	23	333	21	158	20	78
M4 [30]	18	8	16	4	16	7
M5 [31]	20	49	17	12	18	14

Table 2: Time needed for STG generation of the original and the reduced BNs.

Interpretation. In the case of M1 and M2 we could not store the original STG due to our limited computational resources. However, in the case of M1 we could compute the STG of the And-Reduced BN, while for M2 we could compute the reduced STG in both reduction scenarios. For the case of M3, M4, and M5 we have speed-ups. Particularly in the case of M3 STG generation takes more that *5* minutes while in the Or-reduced STG it takes a bit more than *1* minute and in the case of the And-reduced STG *2.5* minutes. To conclude, our reduction method can offer significant speed-ups in the generation of the STG and render the generation feasible despite intractable for the original BN.

D PAPER IV: An Extension of ERODE to Reduce Boolean Networks By Backward Boolean Equivalence

Reproduced with permission from Springer Nature

An Extension of ERODE to Reduce Boolean Networks by Backward Boolean Equivalence *

Georgios Argyris[1 [0000−0002−3203−0410]], Alberto Lluch Lafuente[1 [0000−0001−7405−0818]], Mirco Tribastone[2 [0000−0002−6018−5989]], Max Tschaikowski[3 [0000−0002−6186−8669]], and Andrea Vandin[4,1 [0000−0002−2606−7241]]

[1] DTU Technical University of Denmark, Kongens Lyngby, Denmark
[2] IMT School for Advanced Studies Lucca, Italy
[3] University of Aalborg, Denmark
[4] Sant'Anna School for Advanced Studies, Pisa, Italy

Abstract. Boolean Networks (BN) are established tools for modelling biological systems. However, their analysis is hindered by the state space explosion: the exponentially many states on the variables of a BN. We present an extension of the tool for model reduction ERODE with support for BNs and their reduction with a recent method called Backward Boolean Equivalence (BBE). BBE identifies maximal sets of variables that retain the same value whenever initialized equally. ERODE has been also extended to support importing and exporting between different formats and model repositories, enhancing interoperability with other tools.

Keywords: Boolean Network · Backward Equivalence · Reduction

1 Introduction

Boolean Networks (BNs) [9] are established models for biological systems which have gained a lot of interest due to their simplicity; they consist of Boolean variables which denote active/inactive genes, high/low concentration of substances, etc. The variables are updated according to functions which are encoded into logical rules as we display in the left part of Fig. 1. This BN was published in [6], and models neurogenesis: the process by which nervous system cells, the neurons, are produced by neural stem cells.

A major hurdle in analyzing large BNs is the state space explosion, i.e., the presence of exponentially many *BN states*, the different configurations of (de)activation values of each variable, with respect to the number of BN variables. For example, Fig. 2 shows the state space of the BN of Fig. 1; the BN has 6 variables, leading to 2^6 states. For the tractability of large BNs, several reduction techniques have been proposed (e.g., [1,16,19]). One of the most popular reduction methods is based on fast-slow decomposition, studied in [16,19]

* Partially supported by the DFF project REDUCTO 9040-00224B, the Poul Due Jensen Grant 883901, the Villum Investigator Grant S4OS, and the PRIN project SEDUCE 2017TWRCNB. Corresponding author: Andrea Vandin.

$$
\begin{aligned}
Her6(t+1) &= \neg miR9(t) \wedge \neg N(t) \\
HuC(t+1) &= \neg miR9(t) \wedge \neg P(t) \\
N(t+1) &= HuC(t) \\
P(t+1) &= Her6(t) \vee Zic5(t) \\
Zic5(t+1) &= \neg miR9(t) \wedge \neg N(t) \\
miR9(t+1) &= \neg Her6(t) \wedge \neg N(t)
\end{aligned}
$$

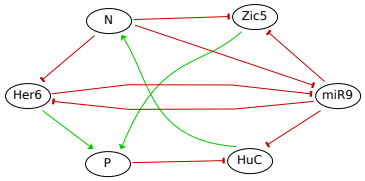


Fig. 1: A BN from [6]. (Left) The variables and update functions. (Right) An abstract graphical representation known as *interaction graph* where the nodes correspond to the variables, and the green/red arrows denote positive/negative effect to the activation value of the target variable, resp.

and implemented in GINsim [5]. Here, we present an extension of ERODE [3], a tool for modelling, analysis, and reduction of biological models that implements a complementary method of reduction for BNs called *Boolean Backward Equivalence* (BBE) [1]. Originally, ERODE was developed to support chemical reaction networks, and systems of ordinary differential equations [3]. Here, we present an extension to support BNs and the new importing and exporting capabilities between three different formats: a native format of ERODE to describe BNs (*.ode*), the *.bnet* format [11], and the *SBML-qual* format [4]. Notably, the latter is a standard for modelling biological systems [5]. These formalisms allow to interface with popular online BN model repositories like BioModelsDB [10] and the GinSim repository [12], as well as tools for BN analysis like those fostered by the COLOMOTO initiative [15].

2 Preliminaries

Model. A BN model is a pair (X, F) with X being a set of variables, and F a set of update functions. In the model of Fig. 1, the set of variables and the set of update functions are: $X = \{Her6, HuC, N, P, Zic5, miR9\}$ and $F = \{f_{Her6}, f_{HuC}, f_N, f_P, f_{Zic5}, f_{miR9}\}$ with, e.g., $f_{Her6} = \neg miR9 \wedge \neg N$.

BBE Partition. The crucial aspect of BBE is the notion of BBE partition (or BBE equivalence), which is a partition of the BN variables that satisfies the following criterion:

> *if the variables within each block have same activation value, they will retain the same value in all subsequent steps.*

An example of partition is $P^1 = \{\{Her6, HuC, N, P, Zic5, miR9\}\}$, which consists of one unique block. Another partition is $P^2 = \{\{Her6, Zic5, P\}, \{HuC\}, \{N\}, \{miR9\}\}$, which consists of four blocks. The partitions P^1, P^2 are not BBE partitions. Instead, $P^3 = \{\{Her6, Zic5\}, \{P\}, \{HuC\}, \{N\}, \{miR9\}\}$ is a BBE partition.

[5] The artifact can be downloaded from `www.erode.eu/examples.html` with further guidelines to replicate the experiments in this document.

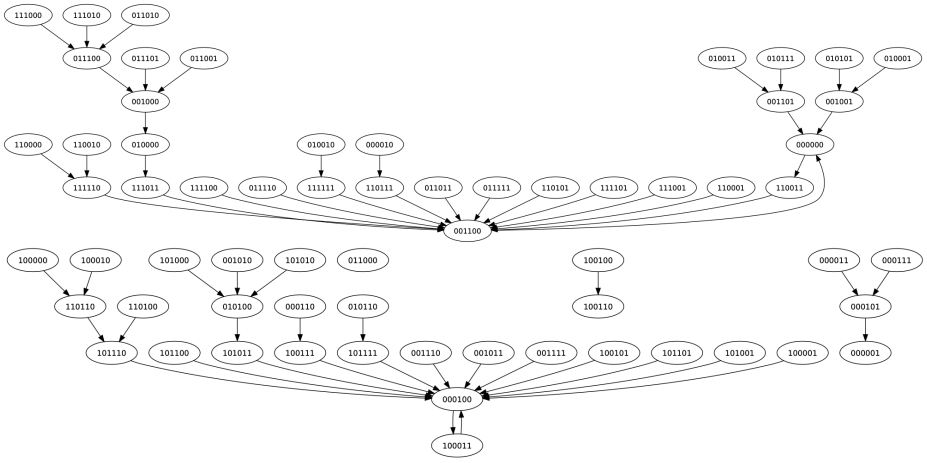


Fig. 2: The state transition graph (STG) of the BN of Fig. 1. An STG encodes the state space (nodes) and the dynamics (transitions) of a BN. The STG consists of 4 disconnected components. Each node contains digits denoting the activation values of each variable in that particular state. The transitions are obtained by synchronously applying the update functions in Fig. 1 to the activation values of the source state.

The BBE reduction method requires the user to specify an initial partition. Following a partition refinement approach [17], BBE proceeds by iteratively splitting the blocks of such partition until a BBE partition is obtained. The maximal BBE reduction of a BN can be obtained by using *trivial* initial partitions with one block only like P^1. By using P^2 as initial partition we get P^3.

Given a BBE partition, we can create a BBE-reduced BN containing one variable per partition block. We have shown in [1] that this can be used to study selected part of the original dynamics.

3 ERODE

Fig. 3 provides a screenshot of ERODE. The middle panel provides the BN of Fig. 1 in ERODE format. The variables shall be declared in a block `begin init ... end init`. We illustrate by comment (`//`), how one could specify initial conditions for some of the variables (set to false by default).

The initial partition for the partition refinement algorithm can be specified in a block `begin partition ... end partition`. In the example of Fig. 3 we declare P^2.

Finally, we declare the update functions for each of the variables in a block `begin update functions ... end update functions`.

After BN definition which is encoded in the previous three blocks, we can provide either reduction or exporting commands. For example, BBE reduction

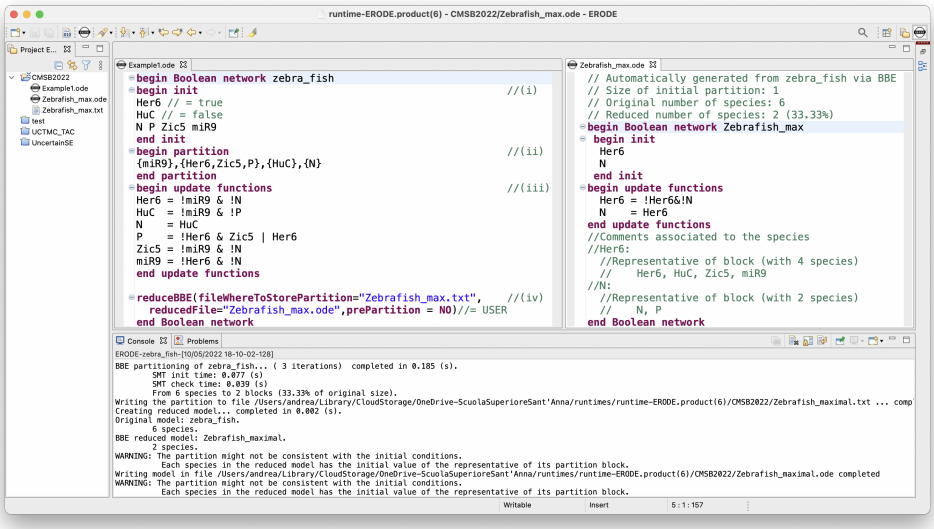


Fig. 3: ERODE GUI: (left) the project explorer; (middle) the BN from Fig. 1 in ERODE format; (right) The BBE reduction of the BN; (bottom) A console with log information. In (right) we see how to specify (i) the variables, (ii) a partition of the variables, (iii) the update functions, and (iv) the reduction commands.

is obtained with command `reduceBBE` which requires 3 parameters. The first, `fileWhereToStorePartition`, names the `.txt` file to store the blocks of the obtained BBE partition. The second parameter, `reducedFile`, names the ERODE file wherein the reduced BN is stored. We display this .ode file in the right window of Fig. 3. The parameter `prePartition` can take three values: `USER` to declare as initial partition the one specified above, `NO` for plugging the trivial partition wherein all variables belong to one block (e.g., P^1), or `IC` to define an initial partition according to the initial conditions specified by the user (one block for all variables initialized to true, and one for those initialized to false).

Before discussing the importing/exporting commands in Section 5, we provide the steps, implemented by ERODE, in our running example.

4 An Illustration of BBE Reduction

In this section, we exemplify how ERODE performs BBE reduction in the BN of Fig. 1. The reduction is summarized in three steps: *(i)* the first step is done by the modeller who provides the BN and an initial partition of the BN variables. ERODE implements the other two steps as follows: *(ii)* it splits the blocks P_i of the initial partition until a partition P that satisfies the BBE criterion is reached. The splitting is done by an iterative partition refinement algorithm [17]. In each iteration, the Z3 SAT solver [7], integrated in ERODE, checks the validity of the

SAT-encoding of the BBE criterion (see [1]); if valid, the current partition is a BBE; if invalid, Z3 returns a counter-example according to which the splitting is performed. Once we get to a partition for which the formula is valid, *(iii)* ERODE produces the reduced BN according to the resulting BBE partition by collapsing all variables that belong to the same block into single variable components.

Partition Refinement. We assume that the modeler sets parameter `prePartition` to `NO`, requiring to use the trivial initial partition P^1. Firstly, the tool checks if P^1 is a BBE partition. Z3 decides that the BBE criterion is invalid which means that P^1 is not a BBE. Z3 provides as counterexample the state $(0,0,0,0,0,0)$, which transits to the state $(1,1,0,0,1,1)$. This means that the third and fourth variable cannot be BBE-equivalent to the others, therefore we refine the initial single block in two separating variables with value 0 and 1. We obtain the new partition: $P = \{\{N, P\}, \{Her6, HuC, Zic5, miR9\}\}$. The algorithm repeats iteratively the above steps until a BBE partition is met. In this case, the algorithm terminates with P because it is a BBE partition.

Reduced BN. When the algorithm reaches a BBE partition, ERODE computes the reduced BN based on it. In the case of the BBE partition $P = \{\{N, P\}, \{Her6, HuC, Zic5, miR9\}\}$, the variables N, P are collapsed into one component $x_{\{N,P\}}$, and the variables $Her6, HuC, Zic5, miR9$ into the variable component $x_{\{Her6,HuC,Zic5,miR9\}}$. The update function of the variable $x_{\{N,P\}}$ is given by selecting the update function of one variable (either N or P), and replacing each occurrence of an original variable with the new one corresponding to its block (i.e., N, and P, are replaced by $x_{\{N,P\}}$, and the others by $x_{\{Her6,HuC,Zic5,miR9\}}$). It can be shown that any update function of the variables in a block can be chosen without affecting the dynamics of the obtained reduced model. In this example we select the variables with the simplest function. We obtain:

$$x_{\{N,P\}}(t+1) = x_{\{Her6,HuC,Zic5,miR9\}}(t)$$
$$x_{\{Her6,HuC,Zic5,miR9\}}(t+1) = \neg x_{\{Her6,HuC,Zic5,miR9\}(t)} \wedge \neg x_{\{N,P\}}(t)$$

We display this BN in the right panel of Fig. 3, where variable $x_{\{N,P\}}$ is denoted by N, and the variable $x_{\{Her6,HuC,Zic5,miR9\}}$ by $Her6$.

Application of BBE reduction for model analysis. Several tasks in model analysis are intractable due to the high dimensionality of BNs e.g., the generation of the STG, and the computation of attractors. Attractors are sets of states towards which the BN tends to evolve and remain. They are usually associated with important behaviours of the underlying system: for instance, different attractors correspond to different cell types in cell differentiation processes [2]. In [1], we present cases wherein BBE reduction can enable of facilitate these tasks.

5 Importing and Exporting Capabilities

Input and Output Variables. The variables of a BN can be divided in 3 categories [14]: *inputs* which denote external stimuli, *outputs* which model readout/response of the modelled system, and internal variables. These categories can be easily observed in the interaction graph of a BN (e.g., Fig. 1 and Fig. 4). Inputs have no incoming edges, outputs have no outgoing edges, and internal variables have both incoming and outgoing edges. ERODE features automatic identification of these three categories basing on the update functions. Input variables can be identified in the BN model as variables that are regulated only by themselves or have a stable update function, i.e. the update function of an input variable x has the form: x, 1, or 0. Instead, output variables do not appear in the update functions of other variables.

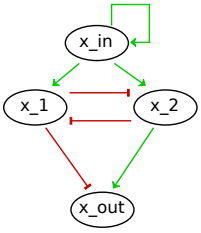


Fig. 4: The interaction graph of a BN with 1 input (x_{in}) and 1 output (x_{out}).

Importing a model. ERODE has importing capabilities from the SBML-qual format [4], which is a standard format for biological models, and the .bnet format. Importing can be done with the following commands:

```
importSBMLQualFolder(folderIn="BNs_sbml",folderOut="BNs_ode")
importBNetFolder(folderIn="BNs_bnet",folderOut="BNs_ode")
```

which load all models from folder `folderIn`, and store the ERODE versions in folder `folderOut`. The commands have an additional optional parameter `guessPrepartition`, triggering the generation of corresponding `partition` block. If set to `outputs`, the outputs are split in singleton blocks, while the others belong to another single block. Similarly for `inputs`. We can also specify `outputsOneBlock` or `inputsOneBlock` in which cases we put all outputs (or inputs) in the same block.

Exporting a model. ERODE can export BNs, e.g. reduced ones, in the above mentioned formats. This is done using commands `exportBoolNet` or `exportSBMLQual`.

6 Conclusion

We extended ERODE to reduce Boolean Networks (BN) with Boolean Backward Equivalence (BBE) which collapses variables such that if initialized equally, retain the same value in all subsequent steps. The scalability and the efficiency the tool has been illustrated in [1] wherein we apply our method to the whole GINsim repository. [6] As future work, ERODE will be extended with further reduction techniques for BNs, complementary to BBE that we presented here. In our future work, we also aim to incorporate ERODE in COLOMOTO notebook [13] to further promote interoperability.

[6] Note to reviewers: The appendix mentions an additional case study material that we will use in showcasing our tool during the conference.

References

1. Argyris, G., Lluch-Lafuente, A., Tribastone, M., Tschaikowski, M., Vandin, A.: Reducing boolean networks with backward boolean equivalence. In: Computational Methods in Systems Biology, CMSB. pp. 1–18 (2021). https://doi.org/10.1007/978-3-030-85633-5_1
2. Azpeitia, E., Benítez, M., Vega, I., Villarreal, C., Alvarez-Buylla, E.R.: Single-cell and coupled grn models of cell patterning in the arabidopsis thaliana root stem cell niche. BMC systems biology **4**(1), 1–19 (2010)
3. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Erode: a tool for the evaluation and reduction of ordinary differential equations. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 310–328. Springer (2017)
4. Chaouiya, C., Bérenguier, D., Keating, S.M., Naldi, A., Van Iersel, M.P., Rodriguez, N., Dräger, A., Büchel, F., Cokelaer, T., Kowal, B., et al.: SBML qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools. BMC systems biology **7**(1), 1–15 (2013)
5. Chaouiya, C., Naldi, A., Thieffry, D.: Logical modelling of gene regulatory networks with ginsim. In: Bacterial Molecular Networks, pp. 463–479. Springer (2012)
6. Coolen, M., Thieffry, D., Drivenes, Ø., Becker, T.S., Bally-Cuif, L.: mir-9 controls the timing of neurogenesis through the direct inhibition of antagonistic factors. Developmental cell **22**(5), 1052–1064 (2012)
7. De Moura, L., Bjørner, N.: Z3: An efficient smt solver. In: International conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 337–340. Springer (2008)
8. Dubrova, E., Teslenko, M.: A sat-based algorithm for finding attractors in synchronous boolean networks. IEEE/ACM transactions on computational biology and bioinformatics **8**(5), 1393–1399 (2011)
9. Kauffman, S.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology **22**(3), 437 – 467 (1969)
10. Malik-Sheriff, R.S., Glont, M., Nguyen, T.V.N., Tiwari, K., Roberts, M.G., Xavier, A., Vu, M.T., Men, J., Maire, M., Kananathan, S., Fairbanks, E.L., Meyer, J.P., Arankalle, C., Varusai, T.M., Knight-Schrijver, V., Li, L., Dueñas-Roca, C., Dass, G., Keating, S.M., Park, Y.M., Buso, N., Rodriguez, N., Hucka, M., Hermjakob, H.: BioModels — 15 years of sharing computational models in life science. Nucleic Acids Research **48**(D1), D407–D415 (1 2020). https://doi.org/10.1093/nar/gkz1055, `https://doi.org/10.1093/nar/gkz1055`, gkz1055
11. Müssel, C., Hopfensitz, M., Kestler, H.A.: Boolnet: an r package for generation, reconstruction and analysis of boolean networks. Bioinformatics **26**(10), 1378–1380 (2010)
12. Naldi, A., Berenguier, D., Fauré, A., Lopez, F., Thieffry, D., Chaouiya, C.: Logical modelling of regulatory networks with ginsim 2.3. Biosystems **97**(2), 134–139 (2009). https://doi.org/https://doi.org/10.1016/j.biosystems.2009.04.008, `https://www.sciencedirect.com/science/article/pii/S0303264709000665`
13. Naldi, A., Hernandez, C., Levy, N., Stoll, G., Monteiro, P.T., Chaouiya, C., Helikar, T., Zinovyev, A., Calzone, L., Cohen-Boulakia, S., et al.: The colomoto interactive notebook: accessible and reproducible computational analyses for qualitative biological networks. Frontiers in physiology **9**, 680 (2018)

14. Naldi, A., Monteiro, P.T., Chaouiya, C.: Efficient handling of large signalling-regulatory networks by focusing on their core control. In: International Conference on Computational Methods in Systems Biology. pp. 288–306. Springer (2012)
15. Naldi, A., Monteiro, P.T., Müssel, C., for Logical Models, C., Tools, Kestler, H.A., Thieffry, D., Xenarios, I., Saez-Rodriguez, J., Helikar, T., Chaouiya, C.: Cooperative development of logical modelling standards and tools with colomoto. Bioinformatics **31**(7), 1154–1159 (2015)
16. Naldi, A., Remy, E., Thieffry, D., Chaouiya, C.: Dynamically consistent reduction of logical regulatory graphs. Theoretical Computer Science **412**(21), 2207–2218 (2011)
17. Paige, R., Tarjan, R.E.: Three partition refinement algorithms. SIAM Journal on Computing **16**(6), 973–989 (1987)
18. Raza, S., Robertson, K.A., Lacaze, P.A., Page, D., Enright, A.J., Ghazal, P., Freeman, T.C.: A logic-based diagram of signalling pathways central to macrophage activation. BMC systems biology **2**(1), 1–15 (2008)
19. Veliz-Cuba, A.: Reduction of boolean network models. Journal of theoretical biology **289**, 167–172 (2011)

A Supplementary Material

We expect to provide a tool demonstration at the conference. This will include some of the experiments presented in [1], and the one presented in this appendix. Notably, the material in the appendix allows us to show that BBE is *useful*, because it allows to analyze models that were intractable before.

A.1 An application to the Signalling Macrophage Activation

In this section we present an application that complements the large set of experiments done in our previous and ongoing studies. We reduce a logic-based diagram of signalling pathways central to macrophage activation [18]. This BN model encompasses four pathways that are central to innate immunity: the toll-like receptor, the inteferon, the NF-κB and apoptotic pathways. The BN consists of *321* variables which refer to proteins, genes or complexes, 19 of them being inputs, 68 of them being outputs and the rest variables are internal. We highlight that the attractors of this BN cannot be computed by using the most efficient tool for attractor identification in synchronous BNs among those supported by the COLOMOTO initiative [8].

Hypothesis. For large models, several tasks are computationally expensive like the computation of attractors. Our crucial hypothesis is that, with BBE reduction, we can compute several attractors which would be otherwise intractable.

Configuration. We explore four different reduction scenarios relevant to input variables: (i) the *Maximal reduction* wherein all variables belong to just one block, (ii) the *Input-separated reduction* wherein the initial partition consists of two blocks (one containing the inputs and one containing the rest variables), (iii)

Input-distinguished reduction wherein all input variables belong to a singleton block of the initial partition, and (iv) the *Manually-refined reduction* where we permit some of the input variables to be merged with other internal variables. The input variables considered in the last case have been selected arbitrarily in order to get results demonstrating that custom initial partitions might allow to handle otherwise untractable models.

Results. The *Input-distinguished reduction* leads to a reduced BN with *161* variables, but, still, attractor computation is infeasible. The *Input-separated* and *Maximal reduction* lead to reduced BNs with *91* and *8* variables respectively, while in these two cases only *2* attractors of the original BN are preserved. However, after arbitrarily selecting some of the inputs, and permitting their merging with internal variables, we obtain a *manually-refined* reduced BN with *137* variables that preserves *8960* attractors of the original system. We summarize the results in Table 1.

Model	*Size*	*Attractors*	*Analysis (s)*	*Reduction (s)*
Original	321	*—Time Out—*		-
Input-distinguished	161	*—Time Out—*		3,295
Manually-refined	137	8960	175,69	3,071
Input-separated	91	2	0.105	3,501
Maximal	8	2	0.032	0,277

Table 1: Different reduction scenarios reproduce different dynamics of the original system.

Interpretation. The *Maximal* and the *Input-separated reduction* usually leads to tractable reduced BNs which, however, may lose several interesting dynamics of the original system. In the case of *Input-distinguished reduction*, we may obtain reduced BNs by several orders of magnitude, but without being able to compute the attractors using state-of-the-art methods [8]. Luckily, there exist cases (like the *Manually-refined reduction*) wherein we reduce enough while preserve more attractors of the original system.