An Extension of ERODE to Reduce Boolean Networks By Backward Boolean Equivalence

Georgios Argyris[1], Alberto Lluch Lafuente[1], Mirco Tribastone[2], Max Tschaikowski[3], and Andrea Vandin[4,1](✉)

[1] DTU Technical University of Denmark, Kongens Lyngby, Denmark
andrea.vandin@santannapisa.it
[2] IMT School for Advanced Studies Lucca, Lucca, Italy
[3] University of Aalborg, 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) [8] 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., [1,15,17]). One of the most popular reduction methods is based

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.

I. Petre and A. Păun (Eds.): CMSB 2022, LNBI 13447, pp. 294–301, 2022.
https://doi.org/10.1007/978-3-031-15034-0_16

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

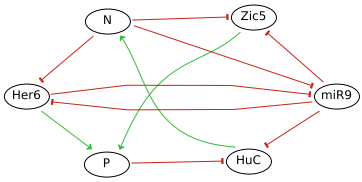

Fig. 1. A BN from [6]. (Left) the variables and update functions. (Right) an abstract graphical representation known as *interaction graph* where the nodes correspond to the variables, and the green/red arrows denote positive/negative effect to the activation value of the target variable, resp. (Color figure online)

on fast-slow decomposition, studied in [15,17] 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 [10], and the *SBML-qual* format [4]. Notably, the latter is a standard for modelling biological systems[1]. These formalisms allow to interface with popular online BN model repositories like BioModelsDB [9] and the GinSim repository [11], as well as tools for BN analysis like those fostered by the COLOMOTO initiative [14].

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.

[1] The artifact can be downloaded from www.erode.eu/examples.html with further guidelines to replicate the experiments in this document.

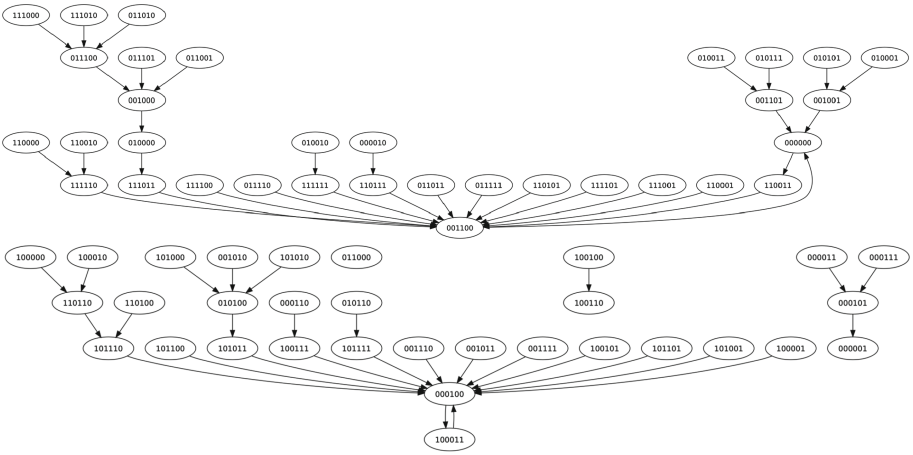

Fig. 2. The state transition graph (STG) of the BN of Fig. 1. An STG encodes the state space (nodes) and the dynamics (transitions) of a BN. The STG consists of 4 disconnected components. Each node contains digits denoting the activation values of each variable in that particular state. The transitions 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 [16], 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

Figure 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 is obtained with command `reduceBBE` which requires 3 parameters. The first, `fileWhereToStorePartition`, names the `.txt` file to store the blocks of

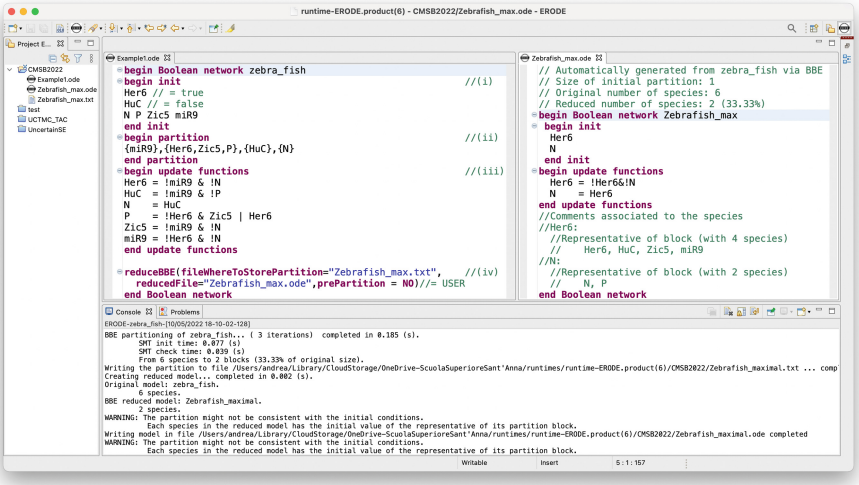

Fig. 3. ERODE GUI: (left) the project explorer; (middle) the BN from Fig. 1 in ERODE format; (right) The BBE reduction of the BN; (bottom) A console with log information. In (right) we see how to specify (i) the variables, (ii) a partition of the variables, (iii) the update functions, and (iv) the reduction commands.

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 Sect. 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 [16]. 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 [13]: *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.

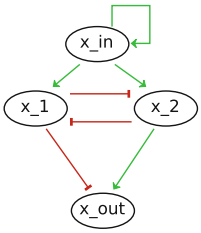

Fig. 4. The interaction graph of a BN with 1 input (x_{in}) and 1 output (x_{out}).

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. 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 [12] to further promote interoperability.

References

1. Argyris, G., Lluch Lafuente, A., Tribastone, M., Tschaikowski, M., Vandin, A.: Reducing Boolean networks with backward Boolean equivalence. In: Cinquemani, E., Paulevé, L. (eds.) CMSB 2021. LNCS, vol. 12881, pp. 1–18. Springer, Cham (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 Syst. Biol. **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: Legay, A., Margaria, T. (eds.) TACAS 2017. LNCS, vol. 10206, pp. 310–328. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54580-5_19
4. Chaouiya, C., et al.: SBML qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools. BMC Syst. Biol. **7**(1), 1–15 (2013)
5. Chaouiya, C., Naldi, A., Thieffry, D.: Logical modelling of gene regulatory networks with GINsim. In: van Helden, J., Toussaint, A., Thieffry, D. (eds.) Bacterial Molecular Networks. Methods in Molecular Biology, vol. 804, pp. 463–479. Springer, New York (2012). https://doi.org/10.1007/978-1-61779-361-5_23
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. Dev. Cell **22**(5), 1052–1064 (2012)
7. de Moura, L., Bjørner, N.: Z3: an efficient SMT solver. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 337–340. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78800-3_24
8. Kauffman, S.: Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. **22**(3), 437–467 (1969)
9. Malik-Sheriff, R.S., et al.: BioModels–15 years of sharing computational models in life science. Nucleic Acids Res. **48**(D1), D407–D415 (2020). https://doi.org/10.1093/nar/gkz1055
10. 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)
11. 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/10.1016/j.biosystems.2009.04.008, https://www.sciencedirect.com/science/article/pii/S0303264709000665
12. Naldi, A., et al.: The CoLoMoTo interactive notebook: accessible and reproducible computational analyses for qualitative biological networks. Front. Physiol. **9**, 680 (2018)
13. Naldi, A., Monteiro, P.T., Chaouiya, C.: Efficient handling of large signalling-regulatory networks by focusing on their core control. In: Gilbert, D., Heiner, M. (eds.) CMSB 2012. LNCS, pp. 288–306. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33636-2_17
14. Naldi, A., et al.: Cooperative development of logical modelling standards and tools with CoLoMoTo. Bioinformatics **31**(7), 1154–1159 (2015)

15. Naldi, A., Remy, E., Thieffry, D., Chaouiya, C.: Dynamically consistent reduction of logical regulatory graphs. Theoret. Comput. Sci. **412**(21), 2207–2218 (2011)
16. Paige, R., Tarjan, R.E.: Three partition refinement algorithms. SIAM J. Comput. **16**(6), 973–989 (1987)
17. Veliz-Cuba, A.: Reduction of Boolean network models. J. Theor. Biol. **289**, 167–172 (2011)