Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 350 (2020) 57–71

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Toward Model Selection by Formal Methods

Matej Hajnal[1] and Samuel Pastva[2,3]

Faculty of Informatics
Masaryk University
Brno, Czech Republic

Abstract

We address the problem of selecting a model from a list of potential models in the field of dynamical systems. The selection is based on model behaviour specified in temporal logic rather than time series. This provides more global constraints on the system dynamics. Not only to select one model but also to create an ordered structure, we propose the model ordering problem. We suggest and apply several ordering relations comparing models given property specification. To provide a formal method with global results for the proposed setting we employ and adapt model checking and parameter synthesis methods. To evaluate the method, we apply the proposed method to several qualitative models of regulatory networks.

Keywords: model checking, parameter synthesis, model selection, FFL

1 Introduction

Models trying to unveil the underlying biological mechanisms are increasing in complexity, size, and number. Thus answering the question of which model represents the system of interest better becomes a challenging task as the models are often incomplete (depicting only a part of the system), heterogeneous (unlike in substance or nature), or mutually inconsistent (contradictory in the network or deducted hypotheses). To address this problem, two approaches are usually used: model discrimination, which excludes unsuitable models; and model selection, the counterpart approach which selects most potential models. Result models serve as a perfect candidate list for further use, prediction making and wet-lab experimental testing, completing the full circle of the standard workflow. Reducing the set of potential models saves the precious time and costs of experimental biologists. In

[1] Email: xhajnal@fi.muni.cz

[2] Email: xpastva@fi.muni.cz

[3] This work has been supported by the Czech Science Foundation grant No. 18-00178S and by the Czech National Infrastructure grant LM2015055. The authors acknowledge Loïc Paulevé et al. for swift work completing one of the used libraries – minibn.

https://doi.org/10.1016/j.entcs.2020.06.004
1571-0661/© 2020 The Author(s). Published by Elsevier B.V.

consequence, a proper understanding of the system and discovery of possible alternations in order to cure undesired behaviour can be achieved faster and cheaper.

In this work, we propose *model ordering*, an approach capable of solving discrimination and selection by creating an ordered structure of models. Further, we present a method for model ordering comparing behavioural aspects of the models as it is the most valuable resource to understand biological mechanisms. In comparison to the standard approaches of model selection [17,18,11], the system behaviour is specified using properties of temporal logic, allowing a more general class of behaviour compared to time series data. Choice of specific logic is not strict as long as it provides adequate reasoning about the system behaviour and sufficiently efficient parameter synthesis procedures. In this work, we employ *computational tree logic* (CTL) – a computationally attractive logic adequately describing non-deterministic behaviour of real dynamical systems.

To verify the behaviour of the models and estimate satisfying parameters we employ model checking techniques focusing on parameter synthesis. The model checking techniques provide global guarantees and time unbounded results overcoming the limitations of approximative, sampling [9], and simulation methods.

To compute an ordering over the set of models, pairs of models are compared based on the given specification by *model ordering relation*. We propose several options for the relations including qualitative measure suitable for division into two groups and also quantitative robustness measures [16]. In this work, we focus on one element of the models – parameter space. For example, the models can be compared by volume or proportion of the satisfying parameter space. Similar orderings may be applied to the state space. The model ordering relations are applicable to any class of specification and modelling formalism underlining generality of the approach.

We examine the applicability of the method by applying the model ordering to models with known behaviour [1] – feed-forward loop (FFL) network motifs. Acquired ordering, an assessment of the quality of the respective network structure, is compared with the real-life incidence. To guarantee the reproducibility, the analysis is available to view at `http://biodivine.fi.muni.cz/paper/SASB2018/FFLs.html` or to edit and run as a jupyter notebook at `http://biodivine.fi.muni.cz/paper/SASB2018/FFLs.ipynb`.

To conclude, model ordering is an approach applicable to many model formalisms, forms of specification, and analysis frameworks. We believe this approach can greatly simplify the model inference workflow while providing a reliable method for evaluating models across formalisms. We are keen on expanding the usability of the proposed method as its reproducibility.

2 Preliminaries

To discriminate, select, or order a set of models, $\mathcal{M}$, the models are compared based on the specification $\mathcal{S}$. To create an ordered structured over the given models, a relation comparing a pair of models is used – *model ordering relation*.

2.1 Model ordering relation

Let $\mathcal{M}$ be a set of models and $\mathcal{S}$ a specification. A model ordering relation, $\leq_{\mathcal{S}}$, is reflexive and transitive binary relation [4] over $\mathcal{M}$.

If a binary relation is also reflexive and transitive then it is a *preorder*. If a preorder relation is also antisymmetric then it is a *partial order*. The partial order allows the possibility to create a *lattice*.

Example 2.1 Consider two arbitrary models – M_1 and M_2. Let specification, $\#var$, denote the number of variables of a model. Let $\leq_{\#var}$, be a model ordering relation which prefers a model with more variables – $M_1 \leq_{\#var} M_2$ iff M_2 has at least as many variables as M_1.

2.2 Problem definition

We consider the following three problems, with model selection and discrimination defined based on the results of model ordering.

- *Model ordering problem* reads as follows: Given a set of arbitrary models $\mathcal{M}$, a specification $\mathcal{S}$, and a model ordering relation $\leq_{\mathcal{S}}$, order the models with respect to $\leq_{\mathcal{S}}$.

- *Model selection problem*: Given models ordered by $\leq_{\mathcal{S}}$, select maximal model(s) with the respect to $\leq_{\mathcal{S}}$.

- *Model discrimination problem*: Given models ordered by $\leq_{\mathcal{S}}$, exclude models lower than a threshold – an element in the ordering relation.

Since we are focused on a behavioural comparison using temporal properties employing model checking techniques, the type of models, specification, and ordering relation applicable to the proposed method are discussed in the following subsections.

2.3 Modelling Formalisms

To harness the advantages of the model checking approach, the model is represented by a *transition system* (TS). To reach this goal, several formalisms of discrete dynamical systems such as Boolean Networks, Petri Nets, and Thomas Networks can be represented in the form of TS by generating the state transition graph. For continuous models a finite discrete abstraction of the state space is necessary. For example, a significant class of ODE (ordinary differential equation) models can be discretised by the rectangular abstraction [4]. The main disadvantage of the abstraction is the fact that it typically leads to over- or under-approximation (or a

[4] a subset of possible pairs – $\mathcal{M} \times \mathcal{M}$

mixture of both) of the dynamics of the original system. A standard TS to represent dynamical systems and employ model checking is a *Kripke structure*.

Definition 2.2 Given AP, a set of atomic propositions, Kripke structure is a tuple $K = (S, S_0, \rightarrow, L)$, where

- S is a finite set of *states*,
- $S_0 \subseteq S$ is a set of initial states,
- $\rightarrow \subseteq S \times S$ is a *transition relation*,
- $L : S \rightarrow 2^{AP}$ is a *labelling function*.

Observing real systems is a challenging task and usually contains an uncertainty – e.g. of its dynamics. Such uncertainty is represented in the model by a parameter, which can be applied in the transition graph by a *parametrised Kripke structure* [3].

Definition 2.3 Given AP, a set of atomic propositions, parametrised Kripke structure is a tuple $K = (S, S_0, P, \rightarrow, L)$, where

- S is a finite set of *states*,
- $S_0 \subseteq S$ is a set of initial states,
- P is a finite set of *parametrisations*,
- $\rightarrow \subseteq S \times P \times S$ is a *transition relation*,
- $L : S \rightarrow 2^{AP}$ is a *labelling function*.

Fixing a parametrisation $p \in P$ reduces the parametrised Kripke structure K to the standard (non-parametrised) Kripke structure $K_p = (S, \xrightarrow{p}, L)$.

This work assumes each parametrisation is given as a valuation of model parameters, such that each parameter is identified using a *parameter index*. The index uniquely identifies the parameter across all considered models. These indices are written using Roman numerals. In Example 3.1, we consider three models with three parameters, indexed as I, II, and III. Here, the first model contains only parameters I and III, the second model contains parameters I and II, and the third model contains parameters II and III.

2.4 Specification

In this work, we compare the models based on their behaviour using temporal properties. To express the properties about the dynamics of systems, we consider formulae of *computational tree logic* (CTL) [8] defined by the following abstract syntax:

$$\varphi ::= Q \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathbf{AX}\varphi \mid \mathbf{EX}\varphi \mid \mathbf{A}(\varphi_1 \,\mathbf{U}\, \varphi_2) \mid \mathbf{E}(\varphi_1 \,\mathbf{U}\, \varphi_2),$$

where Q ranges over *atomic propositions* taken from the set AP. The standard abbreviations $\mathbf{EF}\varphi$ stands for $\mathbf{E}(\mathit{true}\,\mathbf{U}\,\varphi)$ and $\mathbf{AG}\varphi$ means $\neg\mathbf{E}\,\mathbf{F}\,\neg\varphi$. Examples of some typical CTL formulae are [10]:

- **EF** φ expresses reachability of a state where the condition φ holds,
- **AG** φ expresses a stabilisation with φ being continually true,
- $\mathbf{EFAG}\varphi_1 \wedge \mathbf{EFAG}\varphi_2$ expresses a bistable switch (two different stable situations φ_1, φ_2 can be reached).

In order to compare the models with respect to property φ we first have to determine which models satisfy the property. In case of a parametrised model also to find the maximal set from the parameter space for which the property holds. Formally, the model checking problem for Kripke structures reads as follows: Given parametrised Kripke Structure K and temporal property φ is to check whether K meets the property – $K \models \varphi$. The parameter synthesis then refers to a problem: Given parametrised Kripke Structure K and temporal property φ to compute satisfying parameter space, $\bar{P}_\varphi$. In the case of a CTL property φ, $Synth(K, \varphi) = \{p \in P \mid \exists s_0 \in S_0 \ s.t. \ K_p, s_0 \models \varphi\}$.

Remark 2.4 In this paper, we consider a specific variant of the model ordering problem:
Given a set of parametrised Kripke structures, $\mathcal{K}$, a CTL formula φ, and a model ordering relation $\leq_\varphi$, order the models with respect to $\leq_\varphi$.

3 Method

In this section, we propose a method for model ordering of parametrised Kripke structures with respect to temporal properties and ordering relations. As an input, our method assumes a finite set of models represented by parametrised Kripke structures $\mathcal{K} = \{K^m = (S^m, I^m, P^m, \rightarrow^m, L^m) | m \in \{1 \ldots n\}\}$ and a property φ. These structures are then analysed by the parameter synthesis procedure which returns the satisfying parameter subspace $\bar{P}_\varphi$ for each structure K^m. Finally, a chosen ordering relation $\leq_\varphi$ is applied in order to compare the models regarding the results of parameter synthesis, see Figure 1. The variations and extensions of the proposed method are discussed in Section 5.

3.1 Specification

As introduced in Section 2.4 we employ CTL [8], computationally attractive temporal logic adequately describing non-deterministic behaviour of real dynamical systems. Moreover, there are plenty of tools providing model checking and parameter synthesis for CTL [7,13,6,2]. However, choice of specific logic is not strict as long as it provides adequate reasoning about the system behaviour and sufficiently efficient parameter synthesis procedures.

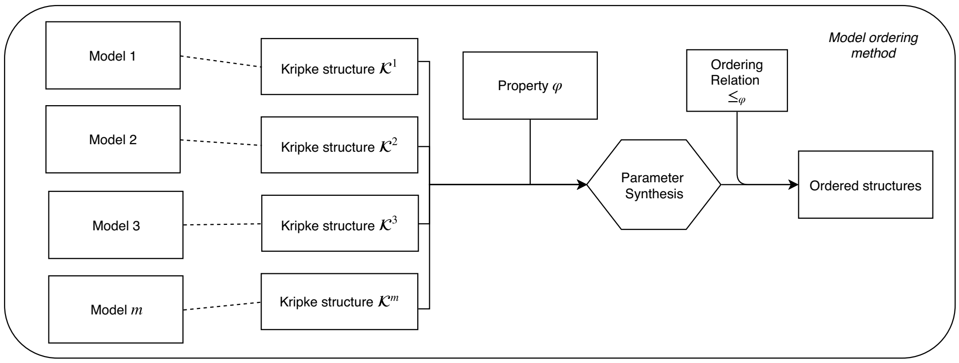
Model 1
Model 2
Model 3
Model m
Kripke structure $\mathcal{K}^1$
Kripke structure $\mathcal{K}^2$
Kripke structure $\mathcal{K}^3$
Kripke structure $\mathcal{K}^m$
Property φ
Parameter Synthesis
Ordering Relation $\leq_\varphi$
Ordered structures
Model ordering method

Fig. 1. Workflow of the method. Input models are represented by parametrised Kripke structures (dashed lines). Parameter synthesis for property φ is run. Given ordering relation $\leq_\varphi$, the structures are ordered.

3.2 Model ordering

An intrinsic method to compare the models against each other based on given specification is to sort them. In this work, we provide several options for model ordering relation given specification in the form of a CTL formula, φ, over the models represented by a Kripke structure, $K \leq_\varphi K'$, with a meaning that K' respects φ at least as good as K. The proposed orderings consider temporal properties focusing on the satisfying parameter subspace for given property, φ, of the individual structure – $\bar{P}_\varphi$.

3.2.1 Model discrimination

Model discrimination requires only qualitative comparison. We offer the following model orderings separating the models by the satisfaction of the given criterion. Ordering employing such division creates two equivalence classes.

(i) Model satisfies the property: $K^m \models \varphi$.
This naive criterion is the cornerstone of the behavioural discrimination. The model is simply excluded if it does not satisfy the property. With introduced parametrisations, more specific representation is needed.

(ii) Exists a parametrisation: $\bar{P}_\varphi^m \neq \emptyset$.
Let us say we observe a property of a system. Therefore the model of the system must contain a parametrisation satisfying the property. The model of the system must satisfy this property for at least one parametrisation. Hence the models without any parametrisation satisfying φ are not suitable.

(iii) For all parametrisations: $\bar{P}_\varphi^m = P^m$.
Let us say we observe a property which is a necessary condition for the system, φ. The model of the system must satisfy this property for each parametrisation. Hence the models with any parametrisation not satisfying φ are not suitable.

3.2.2 Model selection.

Here we provide several quantitative model ordering relations dividing the models into more groups suitable for model selection. Based on the property or the models, one model ordering relation may be more suitable than another. In this section, we propose model ordering relations focusing on parameter space of respective models, while similar orderings may be applied to the state space or structural properties of the model.

The models are primarily compared by the volume or proportion of the satisfying parameter space. However, different models typically contain different parameters, which makes the comparison challenging. To compare such parameter spaces, we propose a restriction of the parametrisations only to the parameters common across all models using a projection π_{com}. This function takes an arbitrary set of parametrisations and restricts each parametrisation to the common parameters. Alternatively, a pairwise projection for two models, $K^m; K^{m'} \in \mathcal{K}$, is also available – $\pi_{com(m,m')}$. Note that in case of non-trivial relationships between the model parameters, a more sophisticated procedure may be necessary to meaningfully unify the parameters.

The provided list of the orderings is not complete. For specific applications, new, more suitable orderings can be proposed. To compare the applicability of individual model ordering relations is a difficult task which we plan to tackle in our future work.

(i) Number of satisfying common parametrisations:
$K^m \leq^1_\varphi K^{m'} \Leftrightarrow |\pi_{com}(\bar{P}^m_\varphi)| \leq |\pi_{com}(\bar{P}^{m'}_\varphi)|$.
A model with more satisfying parametrisations provides higher robustness by means that the property holds for more parameter perturbations.

(ii) Satisfying parametrisations are a subset of common parameters:
$K^m \leq^2_\varphi K^{m'} \Leftrightarrow \pi_{com}(\bar{P}^m_\varphi) \subseteq \pi_{com}(\bar{P}^{m'}_\varphi)$.
$\leq^2_\varphi$ is more strict version of $\leq^1_\varphi$ – $K^m \leq^2_\varphi K^{m'} \Rightarrow K^m \leq^1_\varphi K^{m'}$. It also provides a similar notion of robustness as the previous ordering $\leq^1_\varphi$. The satisfying parameterisation set of the greater structure also contains all parameterisations of the lesser structure, meaning the property holds for at least as many parameter perturbations.
Pairwise common parameters cannot be used to create an order since the ordering relation would not be transitive – see Example 3.1 below.

Example 3.1 A counterexample for transitivity of variations of model orderings $\leq^1_\varphi$ and $\leq^2_\varphi$ with pairwise common parameters on the following three models (Figure 2):

$K^m \leq^{1'}_\varphi K^{m'} \Leftrightarrow |\pi_{com(m,m')}(\bar{P}^m_\varphi)| \leq |\pi_{com(m,m')}(\bar{P}^{m'}_\varphi)|$.
Not transitive: The pairwise common satisfying space of K^1 and K^2 is one parametrisation - (1), for both models hence (K^1, K^2) is in relation. In the case of K^2 and K^3 it is parametrisation (2) hence (K^1, K^2) is also in relation. But (K^1, K^3) is not in the relation, because the common parameter space of

(K^1, K^3) consists of two parametrisations, (4);(5), in the case of K^1 and only one parametrisation in the case of K^3 - (3).
Not antisymmetric: (K^1, K^2) is in relation, but also (K^2, K^1) and $K^1 \neq K^2$.

$K^m \leq^{2'}_{\varphi} K^{m'} \Leftrightarrow \pi_{com(m,m')}(\bar{P}^m_{\varphi}) \subseteq \pi_{com(m,m')}(\bar{P}^{m'}_{\varphi})$
Not transitive: (K^1, K^2) and also (K^2, K^3) are in relation, but (K^1, K^3) is not.
Not antisymmetric: (K^1, K^2) is in relation, but also (K^2, K^1) and $K^1 \neq K^2$.

k	(I, III)
$\bar{P}^1_{\varphi}$	$\{p_1 = (1,4),$ $p_2 = (1,5)\}$

k	(I, II)
$\bar{P}^2_{\varphi}$	$\{p_1 = (1,2)\}$

k	(II, III)
$\bar{P}^3_{\varphi}$	$\{p_1 = (2,3)\}$

Fig. 2. Satisfying parameter subspace of three models. $\bar{P}^m_{\varphi}$ represents satisfying parameter space of model m. p_i indicates parametrisation in the parameter space and k indicates the index of the respective parameter – parameters with the same index are common.

(iii) Number of satisfying parametrisations:
$K^m \leq^3_{\varphi} K^{m'} \Leftrightarrow |\bar{P}^m_{\varphi}| \leq |\bar{P}^{m'}_{\varphi}|$.
Similarly, as in $\leq^1_{\varphi}$, a model with more satisfying parametrisations provides higher robustness by means that the property holds for more parameter perturbations. Since $\leq^3_{\varphi}$ is not strained to common parameters, in the case of uneven parameters of the models the ordering may be unfair (an example in Figure 3).

k	(I, II)
$\bar{P}^1_{\varphi}$	$\{p_1 = (1,2),$ $p_2 = (2,2)\}$

k	(II)
$\bar{P}^2_{\varphi}$	$\{p_1 = (2)\}$

Fig. 3. Satisfying parameter subspace of two models; K^1, K^2; and a property φ. Based on the number of satisfying parametrisations $K^1 \leq^3_{\varphi} K^2$. It is only because the parameter space of the first model contains dimension I with two additional satisfying elements.

(iv) Satisfaction Degree:
$K^m_p \leq^4_{\varphi} K^{m'}_r \Leftrightarrow sd(K^m_p, \varphi) \leq sd(K^{m'}_r, \varphi)$, where $sd(K_p, \varphi))$ is *satisfaction degree* as defined in [9].
Satisfaction degree mimics *evaluation function D* [12] by depicting how well the model preserves property φ for a given parametrisation. This option can be extended for whole parameter space - e.g. by minimal or average satisfaction degree through the parameter space.

(v) Proportion of satisfying parametrisations – Global Robustness:
$K^m \leq^5_{\varphi} K^{m'} \Leftrightarrow R(K^m, \varphi) \leq R(K^{m'}, \varphi)$, where $R(K, \varphi) = \frac{|\bar{P}_{\varphi}|}{|P|}$.

Model with a greater proportion of satisfying state space provides higher global robustness, where the result is normalised by the volume of the whole parameter space instead of a number of parameters as in [11]. The normalisation enables comparison also for distinct dimensions.

(vi) Minimal parametrisation perturbation – Local Robustness: $K_p^m \leq_\varphi^6 K_r^{m'} \Leftrightarrow R(K^m, p, \varphi) \leq R(K^{m'}, r, \varphi)$, where $R(K, p, \varphi) = min(\{dist(p, p') | p' \in P \in K \wedge K_{p'} \nvDash \varphi\})$.
A parametrisation with greater (minimal) distance from not satisfying parameter space provides higher local robustness. Also, this option can be extended for whole parameter space - e.g. by minimal or average distance through the parameter space. Note that in $\leq_\varphi^4$, the is distance measured in the state space and here in the parameter space.

The proposed model ordering relations are not antisymmetric, hence do not create posets. It is because two models can achieve the same characteristic compared by the ordering relation, e.g. set of satisfying parametrisation, while having different transition systems. The poset can be obtained by merging the equivalent models.

4 Case Study

To examine the applicability of the method, we apply the model ordering to models with known behaviour [1] – feed-forward loop (FFL) which is one of the central motifs of gene regulatory networks. There are 13 possible ways to connect three nodes with directed edges. The FFL is the only network motif of the 13 possible three-node patterns. From eight sign combinations of the FFL motif (listed in Figure 4), only two are significant. In the case of the coherent group, it is type 1 – C1-FFL. And for the incoherent group, it is also type 1 – I1-FFL. The six other FFL types seem to appear much less frequently than the C1-FFL and the I1-FLL. [1] To understand the dominance of the two types we examine two dynamic properties of protein Z studied in [1]:

- Sign-sensitive delay. There is a delay of activation of protein Z in switching on the signal, but no delay in switch off and vice versa.
- Pulse generator. In the pulse, Z level first increases and then declines to a low level.

To assess the quality of the competing FFL types, the ordering is computed for each property. The result orderings are compared with the real-life incidence.

4.1 Models and Temporal properties

Firstly, we exhibit the method on the Boolean Network models. Each model contains five variables – three proteins, X, Y, and Z, and two signals, sx and sy, obligatory to turn on the proteins representing enzyme activating specific protein. Each one of the 8 FFL subtypes is modelled with two boolean functions joining two regulations

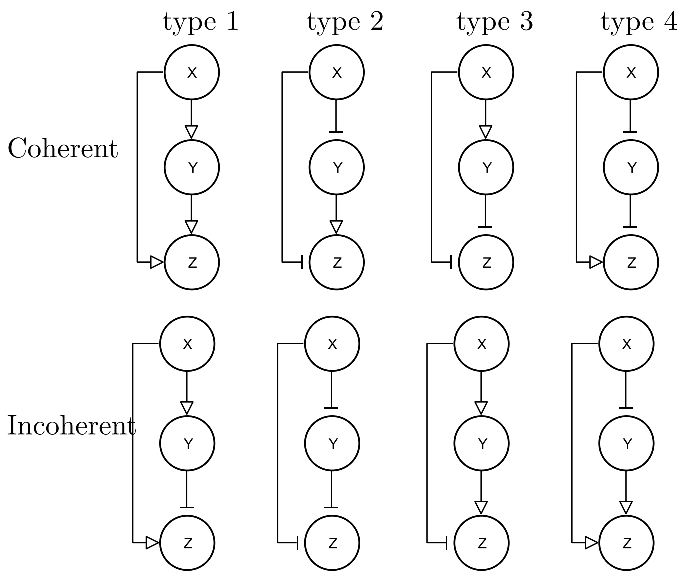

Fig. 4. The eight sign combinations (types) of feed-forward loops. Coherent group possess indirect path with a same overall sign as the direct path. Incoherent, on the other hand, posses indirect path with opposite overall sign to that of the direct path. The arrow symbol $\rightarrow$ denotes activation and the $\dashv$ symbol denotes repression.

for protein Z, AND/OR, creating 16 models.

Secondly, we use Multivalued Network models, which are enhanced with multi-valued protein X and thresholds for regulation of the proteins Y and Z, xy and xz respectively, creating 48 models. [5] To evaluate the method we compare the previous results of [1] with the results of our method.

In the analysis, we have employed model checking using [5] to synthesise parameters one-by-one for the following three properties (with an example of the respective behaviour in Figure 5):
A delay which occurs only in switching on is conjunction of two properties:

$$s_0 = (sx = 1, sy = 1, X = 0, Y = 0, Z = 0), \quad \text{delay\_on} = \mathbf{A}(Z = 0 \ \mathbf{U} \ Y = 1)$$
$$\text{no\_delay\_off} = \mathbf{AF}(((sx = 0 \wedge sy = 1 \wedge Z = 1 \wedge Y = 1) \wedge \mathbf{EX}(Y = 0)) \implies \mathbf{EX}(Y = 0 \wedge Z = 0))$$

delay_on from the initial state s_0 which expresses a delay in switching on the protein Z after the signal appears and no_delay_off – no delay in switching off the protein Z.

$$s_0 = (sx = 0, sy = 1, X = 1, Y = 1, Z = 1), \quad \text{delay\_off} = \mathbf{A}(z = 1 \ \mathbf{U} \ y = 0)$$
$$\text{no\_delay\_on} = \mathbf{AF}(((sx = 1 \wedge sy = 1 \wedge Z = 0 \wedge Y = 0) \wedge \mathbf{EX}(Y = 1)) \implies \mathbf{EX}(Y = 1 \wedge Z = 1))$$

A delay which occurs only in switching off is conjunction of two properties: delay_off from the initial state s_0 which expresses a delay in switching on the protein Z after the signal disappears and no_delay_on – no delay in switching on the protein Z.

[5] we consider 3 cases, $xy < xz$, $xy > xz$, and $xy = xz$

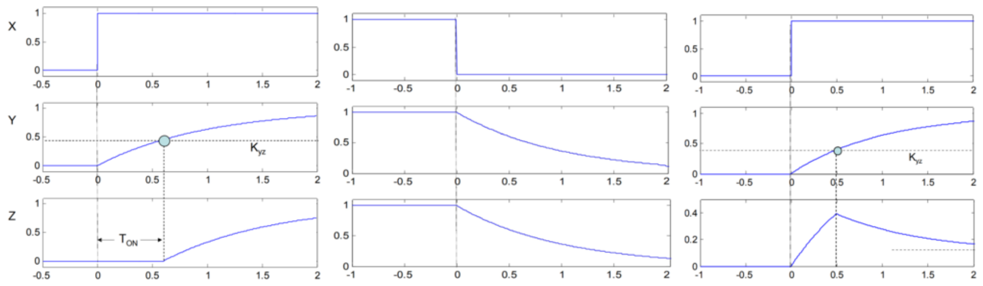

Fig. 5. Example of two behaviours of protein Z – sign-sensitive delay, delay in switching on (left) and no delay in switching off (middle), and pulse generator (right). T_{ON} denotes the length of the delay and k_{yz} denotes the threshold of concentration Y to regulate Z.

AND	C1	C2	C3	C4	I1	I2	I3	I4
$xy = xz$	1		3		3	3	1	3
$xy < xz$	1	3	3			3	1,2	
$xy > xz$	1		3		3	3	1	3

OR	C1	C2	C3	C4	I1	I2	I3	I4
$xy = xz$	2	3	3				2	
$xy < xz$	2	3	2,3				2	
$xy > xz$	2	3	3				2	

Fig. 6. Enumeration of satisfied properties for multivalued models of respective logic – AND (up), OR (down), FFL subtype (column), and threshold setting (row).

$$s_0 = (sx = 1, sy = 1, X = 0, Y = 0, Z = 0), \text{ peak\_gen} = \mathbf{AF}(Z = 1 \land \mathbf{AF}(\mathbf{AG}\, Z = 0))$$

The pulse generator is expressed by peak_gen property.

4.2 Results

The previous analysis [1] unveiled that the C1-AND type satisfies the first property, the C1-OR satisfies the second property, and the I1-AND satisfies the third property. Our analysis uncovered the following results for the BN models:

- In the case of the first property, only for C1-AND and I3-AND subtypes the property holds.
- The second property holds for C1-OR and I3-OR subtypes.
- And the last property holds for I1-AND, C3 both, I4-AND, C2-OR, and I2-AND.

Furthermore, we have analysed multivalued networks using the same properties and initial state with the results shown in Figure 6. The results are compliant with the analysis summarised in [1] since all of the three studied models exhibit the respective property. Also in the case of the third property, models I2-AND and I4-AND can be easily discriminated since the stable state of protein Z is not sensitive to the signal on the protein $Y - sy$ and therefore has reduced functionality. This property was verified by static analysis. However, it is also possible to express it as a temporal property. Since the remaining models satisfy a given property, they serve as the candidate models for a further study – e.g. finding a property explaining their low occurrence.

Using these results, we can easily discriminate and select several models, for example, remove models with no satisfying parametrisation. Furthermore, a model with a superset of satisfying parametrisations in comparison with other models can be selected as more robust.

Since all the models share the same parameter space, the same conclusion can be deduced using the number of satisfying results. Due to the simple nature of our models (small state and parameter space) more advanced orderings are futile.

To view the case study analysis with results use `http://biodivine.fi.muni.cz/paper/SASB2018/FFLs.html`. The analysis is also available as a jupyter notebook at `http://biodivine.fi.muni.cz/paper/SASB2018/FFL.ipynb`. The notebook is editable and runnable using CoLoMoTo docker [14] – image available at `https://hub.docker.com/r/colomoto/colomoto-docker/` with installation instructions. It uses available python libraries for modelling, editing, and analysis of networks – `minibn`, `ginsim`, and `biolqm`. Advantages of the executable notebook are that the models are created on the fly, hence can be easily edited and adapted to analyse other models, and so does the analysis, which can also be run by only selecting parts of it. Moreover, the whole workflow including the results can be presented in a unified form.

4.3 Used frameworks - Jupyter, Docker, and CoLoMoTo-docker

In the analysis we have used following frameworks:

Jupyter (`http://jupyter.org`) provides an interactive web interface for creating documents, named notebooks, which contain code, equations, and formatted texts. A notebook typically describes a full workflow of analysis, both with textual explanations and the full code and parameters to reproduce the results. A notebook is a single file which can be easily modified, shared, re-executed, and visualised online.

Docker (`https://www.docker.com/`) is a computer software that performs operating-system-level virtualisation also known as containerization. Containers are software packages which are isolated from each other and use their own set of tools and libraries; they can communicate through well-defined channels. All containers are run by a single operating system kernel and are therefore more lightweight than virtual machines.

CoLoMoTo-docker (`https://hub.docker.com/r/colomoto/colomoto-docker/`) is a docker image which provides interactive web with a ready-to-use Python environment using Jupyter. It contains libraries aimed on creating, editing, simulation, and analysis of logical networks. It is a joint distribution of several logical modelling software tools easing the chaining of complementary analyses. [15]

5 Discussion

We have presented model ordering as a general concept for comparing models. In this work, we preferred the behavioural comparison to the static attributes and system properties to data. In this section, we discuss three hierarchically different aspects – enhancements and extensions of the current method, variations of the chosen method concerning temporal properties, and an outline of other approaches for model selection while still using system properties rather than data.

5.1 Enhancements and extensions of the current method

The computational bottleneck of the proposed method is the model checking and parameter synthesis procedure due to state space explosion. Moreover, a transition system has to be computed for each model. We propose a way to reduce this computation time.

- Since the models are usually very similar to each other in state space and parameter space, the computation of state transition graph for individual model can be optimized by computing one joint parametrised transition graph.
- The size of the transition system can be in some cases greatly reduced using symbolic data structures.
- Parallel computation – analysis and computation of the transition systems can be often parallelised to better utilise modern multi-core and cloud architectures.

The implementation proposed in this paper is focused on boolean and multivalued networks and properties given in CTL. Next natural step would be to extend the possible applications with new:

- modelling formalisms such as Petri Nets or ordinary differential equations;
- temporal logics such as LTL or STL;
- model ordering relations using more sophisticated parameter unification or domain-specific properties.

5.2 Variations of the approach using temporal properties

- When presented with multiple properties, different approaches to ordering are possible:
  - · Apply chosen ordering relation for each property separately and create an intersection of these relations.
  - · Order the models by the number of satisfied properties, possibly prioritising or weighting the properties.

  - Apply a new specific ordering relation over the separate model orderings.
- Model ordering relations focusing on the state space instead of parameter space.

5.3 Other approaches based on systems properties

There are approaches for model selection/ordering based on dynamical aspects of the system which do not require exact temporal property, thus do not need to be bound with the variables of the system. One of the approaches uses general properties of dynamics and the second compares the dynamics with other systems rather than the property itself.

- General behavioural properties
  - Number of attractors/stable states/etc.
  - Oscillation, plateau, peak generation, liveness, etc.
- System equivalence
  - Bisimulation
  - Topological equivalence (homeomorphism on orbits)
  - Dynamical equivalence (isomorphism)

6 Conclusions

To unveil biological mechanisms, we propose *model ordering approach* in order to reduce the set of candidate models. We start with the problem definition of three problems – model discrimination, selection, and ordering in Section 2.2. Furthermore, a method for model ordering based on the behavioural comparison with focus on the satisfying parameter space is proposed in Section 3. A list of ordering relations with respect to given specification is provided in Section 3.2. Based on this result maximal models can be selected. Finally, the method is applied to a case study of FFL motifs with known behaviour using Boolean and Multivalued Networks in Section 4. The acquired orderings are compared with real-life incidence. The whole analysis is replicable using CoLoMoTo Interactive Notebook [15]. We discuss the possible extensions, adaptation, and variations of the method in Section 5 from which we will work on the following:

- Optimising the method by using parametrised models and structures comprising multiple models.
- Expanding eligible specification by adding other temporal logics.
- Expanding, analysing, and creating methods to compute usable ordering relations.
- As the most important part, we will study the applicability of the method.

References

[1] Alon, U., "An Introduction to Systems Biology: Design Principles of Biological Circuits," Chapman & Hall/CRC Mathematical and Computational Biology, 2006.

[2] Beneš, N., L. Brim, M. Demko, S. Pastva and D. Šafránek, *Pithya: A parallel tool for parameter synthesis of piecewise multi-affine dynamical systems*, in: R. Majumdar and V. Kunčak, editors, *CAV 2017*, LNCS **10426** (2017), pp. 591–598.

[3] Brim, L., M. Demko, M. Češka, S. Pastva and D. Šafránek, *Parameter synthesis by parallel coloured CTL model checking*, in: O. Roux and J. Bourdon, editors, *CMSB 2015*, LNBI **9308** (2015), pp. 251–263.

[4] Brim, L., M. Demko, S. Pastva and D. Šafránek, *High-performance discrete bifurcation analysis for piecewise-affine dynamical systems*, in: A. Abate and D. Šafránek, editors, *Hybrid Systems Biology 2015*, LNBI **9271** (2015), pp. 58–74.

[5] Cimatti, A., E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani and A. Tacchella, *NuSMV Version 2: An OpenSource Tool for Symbolic Model Checking*, in: *Proc. International Conference on Computer-Aided Verification (CAV)*, LNCS **2404** (2002).

[6] Cimatti, A., E. Clarke, F. Giunchiglia and M. Roveri, *Nusmv: A new symbolic model verifier*, in: *International conference on computer aided verification*, Springer, 1999, pp. 495–499.

[7] Clarke, E. M. and E. A. Emerson, *Design and synthesis of synchronization skeletons using branching time temporal logic*, in: *Workshop on Logic of Programs*, Springer, 1981, pp. 52–71.

[8] Clarke, E. M., E. A. Emerson and A. P. Sistla, *Automatic verification of finite-state concurrent systems using temporal logic specifications*, ACM Trans. Program. Lang. Syst. **8** (1986), pp. 244–263.

[9] Fages, F. and A. Rizk, *From model-checking to temporal logic constraint solving*, in: *International Conference on Principles and Practice of Constraint Programming*, Springer, 2009, pp. 319–334.

[10] Fages, F. and S. Soliman, *Formal cell biology in biocham*, in: *SFM*, LNCS **5016** (2008), pp. 54–80.

[11] Hafner, M., H. Koeppl, M. Hasler and A. Wagner, *'Glocal' robustness analysis and model discrimination for circadian oscillators*, PLoS computational biology **5** (2009), p. e1000534.

[12] Kitano, H., *Towards a theory of biological robustness*, Molecular systems biology **3** (2007), p. 137.

[13] McMillan, K. L., *Symbolic model checking*, in: *Symbolic Model Checking*, Springer, 1993 pp. 25–60.

[14] Naldi, A., C. Hernandez, N. Levy, G. Stoll, P. T. Monteiro, C. Chaouiya, T. Helikar, A. Zinovyev, L. Calzone, S. Cohen-Boulakia, D. Thieffry and L. Paulevé, *The CoLoMoTo Interactive Notebook: Accessible and Reproducible Computational Analyses for Qualitative Biological Networks*, bioRxiv (2018).

[15] Naldi, A., 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 **31** (2015), pp. 1154–1159.

[16] Rizk, A., G. Batt, F. Fages and S. Soliman, *A general computational method for robustness analysis with applications to synthetic gene networks*, Bioinformatics **25** (2009).

[17] Silk, D., P. D. Kirk, C. P. Barnes, T. Toni and M. P. Stumpf, *Model selection in systems biology depends on experimental design*, PLoS computational biology **10** (2014), p. e1003650.

[18] Toni, T. and M. P. Stumpf, *Simulation-based model selection for dynamical systems in systems and population biology*, Bioinformatics **26** (2009), pp. 104–110.