rPrism – A software for reactive weighted state transition models

Daniel Figueiredo[1], Eugénio Rocha[1], Manuel António Martins[1], Madalena Chaves[2]

[1]CIDMA – University of Aveiro, Portugal
[2]INRIA – Sophia Antipolis, Mediterranée, France

Abstract

In this work we introduce the software rPrism, as a branch of the software *PRISM model checker*, in order to be able to study weighted reactive state transition models. This kind of model gathers together the concepts of reactivity – which consists of the capacity of a state transition model to alter its accessibility relation – and weights, which can be seen as costs, rates, *etc.*. Given a specific model, the tool performs a simulation based on a Continuous Time Markov Chain. In particular, we show an example of its application for biological systems.

Keywords: *rPrism*; *PRISM model checker*; Reactive models; Weighted switch graphs

1 Introduction

The concept of reactivity in state transition models have been introduced by several authors such as van Benthem, Areces and Gabbay and some examples can be found in [1, 2, 3, 8, 9]. These reactive models are those whose accessibility relation (set of edges) is not fixed but can vary according to a taken path. In some sense, it can be seen as a model with memory.

The authors mentioned before, proposed several formalisms to study such models. In this paper, we will focus on the approach of [8]. In this paper, switch graphs are presented and their application is illustrated with some examples. In particular, systems whose dynamics can be described using counters or demanding some specific order to evolve are shown to be more efficiently described by reactive models. Also in biology, this kind of model can be applied: a previous work in a related topic, where a reactive model is proposed for the study of biological regulatory networks can be found in [6].

Here, we present the tool rPrism that was designed as a branch of Prism model checker ([10]) to study such reactive models. The proposed tool calls PRISM to simulate the evolution of a reactive state transition model.

1.1 Background

We start by introducing some theoretical foundations in the topic in order to be able to explain better the relevance and usability of rPrism.

Definition 1 (Switch graph)

A switch graph is a pair (W, S) such that W is a non empty set of states and S is defined recursively as:

- $S_0 \subseteq W \times W$;
- $S_{i+1} \subseteq S_0 \times S_i \times \{\circ, \bullet\}$, for $i \in \mathbb{Z}_0^+$;
- $S = \bigcup_{i \in \mathbb{Z}_0^+} S_i$.

We say that S is the set of edges and the edges in $S \backslash S_0$ are higher-level edges. Furthermore, if $e \in S_i$, e is said to be a i-level edge. Also, the initial configuration of a switch graph is given by an initial *instantiation* function $I_0 : W \to \{0, 1\}$.

A higher-level edge $(d, e, *)$ is said to be an activator if $*$ is $\bullet$ and is said to be an inhibitor if $*$ is $\circ$. It means that it will either inhibit (temporarily remove) or activate (reintroduce) its target edge e in the model whenever the source edge d is crossed. If the state of the target edge already agrees with the directed by the higher-level edge $(d, e, *)$, then it has no effect.

In the graphical representation of a switch graph, as shown in Fig. 1, inhibitor edges are depicted as white headed arrows, while black headed arrows represent activator edges.

At any time, the configuration of a switch graph is given through an *instantiation* function $I : S \to \{0, 1\}$ which marks each edge as inhibited (temporarily removed) or active depending on $I(s)$. An edge is active if $I(s) = 1$ and it is inhibited otherwise. The former edges are depicted as dashed arrows while the later edges are depicted as full arrows. Inhibited edges cannot be crossed, and they can neither activate nor inhibit other edges. Moreover, only 0-level edges (going from nodes to nodes) can be crossed: if one such edge x is crossed, all active higher-level edges with source in x, *i.e.* $(x, e, *)$ will fire and activate/inhibit the respective target edge e.

Example 1
Fig 1 depicts a switch graph (W, S) with $W = \{w\}$ and

$$S = \{(w, w), \big((w, w), (w, w), \circ\big), \big((w, w), \big((w, w), (w, w), \circ\big)\big), \bullet\big)\}$$

For simplicity, we define $e_1 = \big((w, w), \big((w, w), \circ\big)\big), \bullet\big)$ and $e_2 = \big((w, w), (w, w), \circ\big)$. The initial instantiation I_0 is such that $I_0(w, w) = 1$ (the edge (w, w) can be crossed), $I_0(e_2) = 0$ (meaning that it is inhibited) and $I_0(e_1) = 1$ (therefore, activated and ready to activate e_2, the pointed edge whenever (w, w) is crossed). Therefore, starting from w, the edge (w, w) can be crossed (since it is active) and this causes the higher-level edge e_1 to fire and activate e_2. But e_2 has no effect since it was originally inhibited when (w, w) was crossed. One can then cross (w, w) again. Now, e_1 acts but has no effect, since e_2 is already active, while e_2 acts and inhibits (w, w). Hence, (w, w) can no longer be crossed. This is illustrated in Fig. 1.

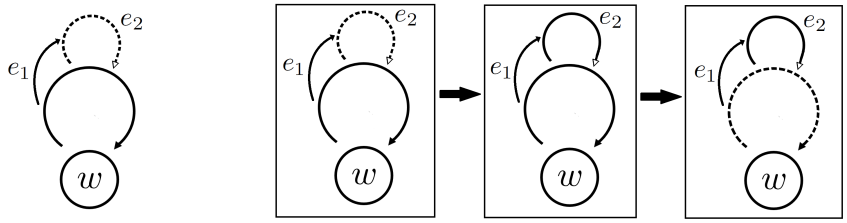

Figure 1: Example and evolution of a switch graph.

Switch graphs can indeed be used in several fields. See [6, 7] for more examples.

1.2 Weighted switch graphs

In this section, we introduced a generalization of switch graphs to include weights. Weights in state transition models are very useful and can represent diverse mechanisms such as costs, distances, rewards, probabilities, besides others.

Definition 2
A *weighted switch graph* is a pair (W, S) together with an initial *instantiation* $I_0 : S \to \Omega \cup \{\circledcirc\}$ where Ω is the set of weights, and can be chosen according to the context.

In weighted switch graphs, instead of simply considering that an edge is active or inhibited, each domain has a weight. In order to express this, we can generalize the notion of instantiation. This is attained by considering an instantiation as a function whose images belong to a set of weights Ω, along with an element $\circledcirc$ as image. Thus, if s is an edge of the model and I an instantiation, $I(s) = \circledcirc$ mean that the edge s is inhibited (temporarily removed from the model) and, otherwise, we say that the edge s is active and with weight $I(s)$.

Given this definition, we can describe the evolution of a weighted switch graph when some edge $s \in W \times W$ is crossed in the following way:

$$I^+(t) = \begin{cases} I(t), & \text{if } (s,t,*) \notin S \ \vee \ I\big((s,t,*)\big) = \circledcirc, \text{ for any } * \in \{\bullet, \circ\} \\ \circledcirc, & \text{if } (s,t,\circ) \in S \text{ and } I\big((s,t,\circ)\big) \neq \circledcirc \\ I\big((s,t,\bullet)\big), & \text{otherwise.} \end{cases}$$

Although we introduced this general definition, we only consider a particular class of models for now. The current version of the package rPrism, version 1.0, is suitable for one-level weighted switch graphs, which are defined as follows.

Definition 3
A *one-level weighted switch graph* is a pair (W, S) with $W \neq \{\}$ and $S = S_0 \cup S_1$ such that:

- $S_0 \subseteq W \times W$
- $S_1 \subseteq S_0 \times S_0$

along with an initial *instantiation* $I_0 : S \to \Omega$, where Ω is the set of weights.

Note. In this context, $\circledcirc \equiv 0$, because our weights will be conceived as rates.

2 About the tool rPrism

As mentioned before, rPrism is suitable to deal with one-level weighted switch graph. Furthermore, the codomain for the considered instantiations must be $\mathbb{Q}_0^+$ and, in practical cases, weights should be considered as rates. For instance, if an edge from a node w to a node w' has weight a, then it means that the component represented by w will become the component w' with rate a. The tool then performs a stochastic simulation based on a Continuous Time Markov Chain.

Given a one-level weighted switch graph, the user of rPrism must specify the model in a simple text format, with the following structure:

```
NS {
 N "definition of node 1" {
   "definition of edge 1";
   "definition of edge 2";
   ...
 }
 N "definition of node 2" {
 }
 ...
}

H1 {
  "definition of one-level edge 1";
  "definition of one-level edge 2";
  ...
}

options "command1";
output  "command2";
sim cmtc;
```

The "definition of a node" has the following format:

"node label" "lbound" "ubound" "initvalue"

where "node label" is a valid string with the name/identifier of the node; "lbound" (respectively, "ubound") is an integer determining the lower (respectively, upper) bound with respect to the number of elements of type "node" on

the system; and “initvalue” is a integer with the initial value of elements of type “node” on the system.

The “definition of an edge” is done in the following way:

”target node” ”initial weight”

where “target node” is a valid string with the name/identifier of the target node; and “initial weight” is a float with the initial weight of the edge.

Finally, to define a one-level edge, we must use the following code:

”source edge” ”target edge” ”weight”

where both “source edge” and “target edge” are strings and have the format “source node”:“target node”; and “weight” is a float with the weight of the one-level edge.

The entry “command1” determines the output of the program and must be filled according to the goal of the user, for example, can be ”simpath 10” (meaning the simulation will make at least 10 steps) or ”simtime 5.7” (the simulation will run at least until the unit of time reaches the value 5.7). The entry ”command2” can be replaced as “all” in order to obtain the entire set of outputs or restricted to any combination of the commands: “odel”, “simulation_results”, “simulation_plot”, “reachable_sets”, “transition_matrix”, “labels”, separated by a space.

Given a one-level switch graph, rPrism translates the introduced model into PRISM language in order to use it to study reactive models. For readers who are used to PRISM syntax, the process traduces nodes to variables and edges to actions. However, an additional module for rates is considered. There, higher-level edges are encoded as an additional variable whose value determines the rates of target edges (*actions*).

Finally, we point out that the rPrism software is implemented as a sDL package[1], and an online demo is available[2] for testing purposes. Nevertheless, the online demo has several limitations due to the fact of being over a web browser, such specific limitations do not exist when using directly the sDL client.

3 Modeling biochemical systems

Switch graphs can describe diverse dynamics of systems which regular graphs can not. An example is the possibility of describing counters such as the one illustrated at Fig. 1 and many others can be found in [3, 7, 9, 8]. Also in biochemical contexts, we can find systems which can intuitively be described by switch graphs. Indeed, as mentioned before, an application of switch graph to the study of biological regulatory networks can be found in [6].

A simple example of a biochemical process which can be modeled using a switch graph is presented in Fig. 2: the scheme represents the general sequence

[1] http://sdl.mathdir.org

[2] http://sdl-vm2.mathdir.org/demos/sDL-pck-run?pck=rPrism/1.0&sdoc=Example_A

of stages related to a vaccination process. Upon vaccination, a susceptible individual immediately has a lower probability of becoming infected if it comes into contact with a virus. This fact is described by the inhibition arrow.

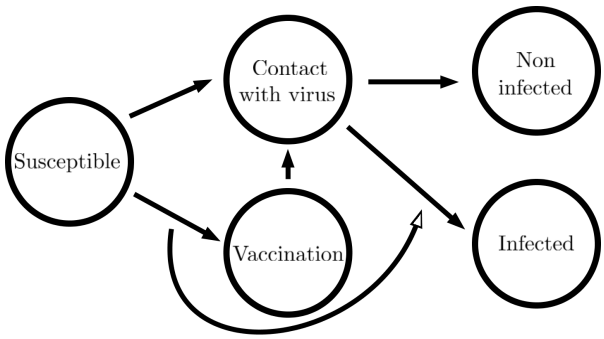

Figure 2: Example of a switch graph describing a biological systems.

Another biological system which could be described by reactive formalisms is the *cooperativity of a hemoglobin protein* and it is described in [5]. In this system, a hemoglobin protein can bind up to 4 oxygen molecules. Thus, although this seems a simple systems, it needs to be described by a model which accommodates features such as counters, and switch graphs are perfect for the case. However, at this point, we must note that even switch graphs do not fully describe the dynamics of the mentioned system. Indeed, the cooperativity of hemoglobin is characterized by the fact that binding to one oxygen molecule increases the likelihood of binding to another one (up to the maximum of four). This increase of the binding rate cannot be described by simple switch graphs which do not consider any kind of quantitative measure. Thus, this issue is solved with weighted switch graphs which admit weights in edges.

The example of a weighted switch graph describing a hemoglobin protein is shown in Fig. 3. There, each loop represents the binding of one oxygen molecule and the weight of the loop represents the respective rate: note that the weights increase for the binding of sucessive oxygen molecules, but a fifth molecule can no longer bind.

Finally, we introduce another example and show how rPrism can be used to study weighted switch graph models.

Example 2 (Circadian rhythm of cyanobacteria)

In [4] we may find a model for the circadian rhythm of a cyanobacteria which considers three phosphorylated forms of the protein KaiC (s, t and ts) and an unphosporylated form (u). Here, we omit the occurrences of protein KaiA, as related in the model of [4], in order to show that it can be represented by the reactivity of the system. In fact, a one-level weighted switch graph for such model is presented in Fig.4.

rPrism was used to simulate the evolution of this system. The plot of the output is presented in Fig. 5. Below, the code introduced in rPrism is presented.

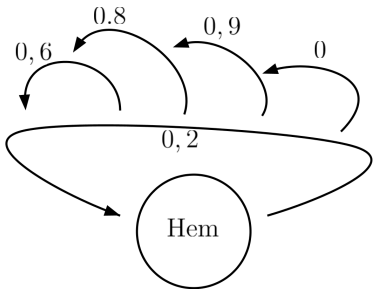

Figure 3: Model representing the cooperativity of hemoglobin.

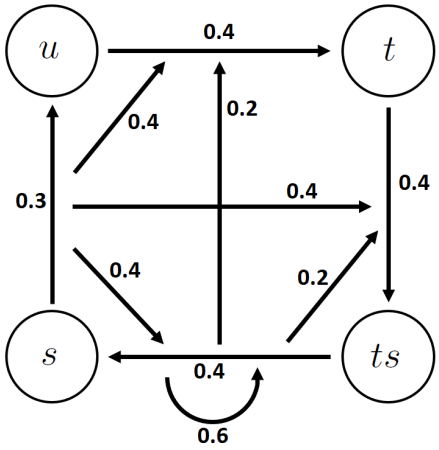

Figure 4: Weighted switch graph of the biological circadian rhythm model.

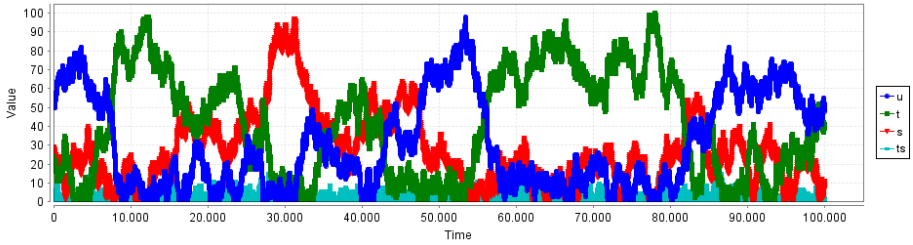

Figure 5: Plot of the simulation.

We note that this model shows a cyclic behavior as would be expected for a circadian rhythm system. Thus, this kind of model seems suitable to represent by higher-level edges the effect of non linear or even unknown mechanisms of a cell and still obtain coherent simulations and results.

Through all examples we can find a common pattern: reactivity and, in particular, higher-level edges appear to describe the dynamics caused by a component/variable which is not considered. Indeed, note that, for instance, in the hemoglobin example we could consider a model with five states where each node is fully described by the number of oxygen molecules bound. In this way, we could obtain a model as the one shown in Fig. 6. Compared to the this one, the reactive model in Fig. 3 ignores a variable: the number of oxygen molecules already bound by the hemoglobin protein.

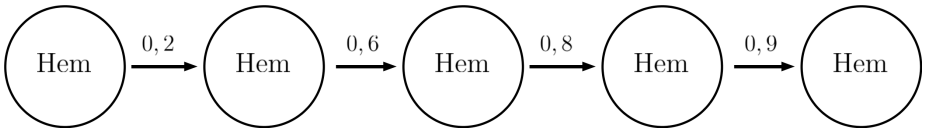

Figure 6: Non-reactive weighted model for hemoglobin protein.

We note that the proposed software – rPrism – is able to, in general, construct a reactive model which contains less states than a non-reactive model. Nevertheless, rPrism internally introduces additional state variables when translating the introduced model into PRISM language, in order to retrieve the hidden information about the system. Using simple words, rPrism considers additional variables which determine what weights must be considered for each edge, at each time. Therefore, we cannot think about one-level switch graphs as reduced models but as a different description of the same model with, in general, the same "computational size". In this way, weighted switch graphs and rPrism are specially useful in two general cases:

- When the user understands that the system is more intuitively described by a reactive model, which in general depends on the background of the user.
- When the "hidden" components/variables causing reactivity are still unknown by the user.

In fact, the second point described above occurs frequently in biological contexts when, for instance, there is a missing or misunderstood regulation between two components in a system. Reactive formalisms allows one to, even so, recover coherent results from such model.

4 Conclusions and future work

In this paper we briefly introduce the software rPrism to study one-level weighted switch graphs. The proposed software has a proper syntax in order to be a friendly software, translating the rPrism input language to a suitable input for simulations in Prism. Also, we present an example of a biological system which can be modeled and studied using this approach.

As future work, we intend to extend the rPrism language with the aim of exploring as much as possible the rich set of features of PRISM, namely, adding some model checking capabilities for reactive graphs. In fact, we choose to use PRISM to be the basis of our work based on its model checking capacities. It has a temporal logic language embedded and it would allow us to compute the validity of properties as well of probabilities for some events to occur, which would be quite relevant for model analysis and predictions for biological systems. Also, we intend to expand the class of models that are suitable to be studied using rPrism. In particular, we intend to consider the general set of higher-level edges. Finally, as ongoing work, we are applying rPrism to a wider number of biological problems.

Acknowledgments.

This work was supported by ERDF - The European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project POCI-01-0145-FEDER-030947 and project with reference UID/MAT/04106/2019 at CIDMA. The authors acknowledge the support given by a France-Portugal partnership PHC PESSOA 2018 between M. Chaves (Campus France #40823SD) and M. A. Martins. D. Figueiredo also acknowledges the support given by FCT via the PhD scholarship PD/BD/114186/2016.

References

[1] Areces, C., Fervari, R., Hoffmann, G.: Swap logic. Logic Journal of IGPL p. jzt030 (2013)

[2] Areces, C., Fervari, R., Hoffmann, G.: Relation-changing modal operators. Logic Journal of IGPL p. jzv020 (2015)

[3] van Benthem, J.: An essay on sabotage and obstruction. In: Mechanizing Mathematical Reasoning, pp. 268–276. Springer (2005)

[4] Chaves, M., Preto, M.: Hierarchy of models: From qualitative to quantitative analysis of circadian rhythms in cyanobacteria. Chaos: An Interdisciplinary Journal of Nonlinear Science 23(2), 025113 (2013)

[5] Chou, K.C.: Low-frequency resonance and cooperativity of hemoglobin. Trends in Biochemical Sciences 14(6), 212 (1989)

[6] Figueiredo, D., Barbosa, L.S.: Reactive models for biological regulatory networks. In: International Symposium on Molecular Logic and Computational Synthetic Biology. p. (to be published). Springer (2019)

[7] Figueiredo, D., Martins, M.A., Barbosa, L.S.: A note on reactive transitions and Reo connectors. In: It's All About Coordination, pp. 57–67. Springer (2018)

[8] Gabbay, D., Marcelino, S.: Global view on reactivity: switch graphs and their logics. Annals of Mathematics and Artificial Intelligence 66(1-4), 131–162 (2012)

[9] Gabbay, D.M., Marcelino, S.: Modal logics of reactive frames. Studia Logica 93(2), 405–446 (2009)

[10] Kwiatkowska, M., Norman, G., Parker, D.: Prism 4.0: Verification of probabilistic real-time systems. In: International conference on computer aided verification. pp. 585–591. Springer (2011)

A Appendix

The code used in rPrism for the circadian rhythm reactive model is presented bellow:

```
NS {
N s  0 100 25 {
u  0.3;
}
N ts 0 100 25 {
s  0.4;
}
N t  0 100 25 {
ts 0.4;
}
N u  0 100 25 {
t  0.4;
}
}

H1 {
s:u  u:t  0.4;
s:u  ts:s 0.4;
s:u  t:ts 0.4;
ts:s u:t  0.2;
ts:s t:ts 0.2;
ts:s ts:s 0.6;
}

options simtime 100000;
output  all;
sim cmtc;
```