Model Checking Genetic Regulatory Networks using GNA and CADP

Grégory Batt, Damien Bergamini, Hidde de Jong,
Hubert Garavel, and Radu Mateescu

INRIA Rhône-Alpes, 655, avenue de l'Europe, Montbonnot,
F-38334 Saint Ismier Cedex, France
{Gregory.Batt,Damien.Bergamini,Hidde.de-Jong,Hubert.Garavel,Radu.Mateescu}
@inria.fr

1 Introduction

The study of genetic regulatory networks, which underlie the functioning of living organisms, has received a major impetus from the recent development of high-throughput genomic techniques. This experimental progress calls for the development of appropriate computer tools supporting the analysis of genetic regulatory processes. We have developed a modeling and simulation method [5, 7], based on piecewise-linear differential equations, that is well-adapted to the qualitative nature of most available biological data. The method has been implemented in the tool Genetic Network Analyzer (GNA) [6], which produces a graph of qualitative states and transitions between qualitative states. The graph provides a discrete abstraction of the dynamics of the system.

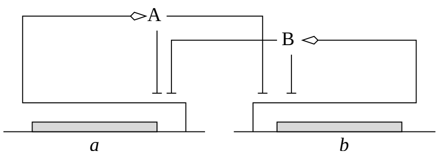
A bottleneck in the application of the qualitative simulation method is the analysis of the state transition graph, which is usually too large for visual inspection. In this paper, we propose a model-checking approach to perform this task in a systematic and efficient way. Given that certain properties of biological interest are of a branching nature (see, *e.g.*, the *bistability* property in Section 3), a branching-time temporal logic is necessary. Also, abstractions of state transition graphs can be performed more conveniently by using standard equivalence relations defined on Labeled Transition Systems (LTSs) rather than by implementing *ad hoc* reductions. Therefore, we developed a connection between the qualitative simulator GNA and the widely-used CADP verification toolbox [8], which provides the required analysis functionalities on LTSs.

The connection is established as follows. Firstly, a dedicated translator converts the state transition graph resulting from qualitative simulation into an LTS suitable for automated verification. Then, after instantaneous states have been abstracted away by means of branching bisimulation, various properties characterizing the evolution of protein concentrations are checked by encoding them in regular alternation-free μ-calculus. The diagnostics produced by the CADP model checker make it possible to establish a correspondence between verification results and biological data, for instance by characterizing evolutions leading to equilibrium states. We illustrate the combined use of qualitative simulation and model checking by means of a simple, biologically-inspired example.

2 Qualitative Simulation of Genetic Regulatory Networks

We consider qualitative models of genetic regulatory networks, based on a class of piecewise-linear differential equations originally proposed in mathemetical biology [10]. Given a qualitative model of a genetic regulatory network, the qualitative simulation method produces a graph of qualitative states and transitions between qualitative states, qualitatively summarizing the dynamics of the system [5, 7]. In the sequel, we present the method by means of an example.

Figure 1(a) represents a simple genetic regulatory network consisting of two genes, a and b, and two proteins, A and B. When a gene is expressed, the corresponding protein is synthesized, which, in turn, can regulate the expression of its own and the other gene. For example, when gene a is expressed, protein A is synthesized and, depending on whether its concentration is above or below a threshold, it may inhibit the expression of gene a and/or b. This network can be described by means of the differential equations (1)-(2), where x_a and x_b denote the concentration of proteins A and B, θ_a^1, θ_a^2, θ_b^1, and θ_b^2, threshold concentrations and s^-, the decreasing step function. For example, equation (1) states that protein A is produced (at a rate κ_a), if and only if $s^-(x_a, \theta_a^2)\, s^-(x_b, \theta_b^1) = 1$, that is, if and only if x_a and x_b are below thresholds θ_a^2 and θ_b^1 respectively. In addition, protein A is degraded at a rate proportional to its own concentration ($\gamma_a > 0$). The parameter inequalities (3)-(4) constrain the parameter values.



(a)

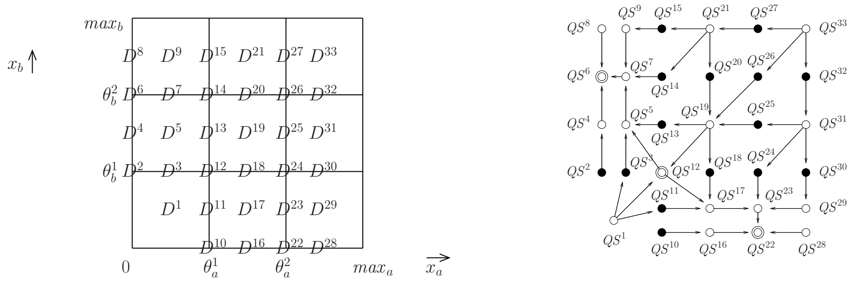
$$\dot{x}_a = \kappa_a\, s^-(x_a, \theta_a^2)\, s^-(x_b, \theta_b^1) - \gamma_a\, x_a \quad (1)$$
$$\dot{x}_b = \kappa_b\, s^-(x_a, \theta_a^1)\, s^-(x_b, \theta_b^2) - \gamma_b\, x_b \quad (2)$$
$$0 < \theta_a^1 < \theta_a^2 < \kappa_a/\gamma_a < max_a \quad (3)$$
$$0 < \theta_b^1 < \theta_b^2 < \kappa_b/\gamma_b < max_b \quad (4)$$

(b)

Fig. 1. (a) Example of a genetic regulatory network of two genes, a and b. The notation follows, in a somewhat simplified form, the graphical conventions proposed by Kohn [11]. (b) Qualitative model, corresponding to the two-gene example, composed of piecewise-linear differential equations (1)-(2) and parameter inequalities (3)-(4).

The phase space can be partitioned into (hyper)rectangular regions, called *flow domains*, where the flow is qualitatively identical, that is, where the sign of the derivatives is identical for all solutions (see Figure 2(a)). For example, in flow domain $D^1 = [0, \theta_a^1[\times [0, \theta_b^1[$, the expression $s^-(x_a, \theta_a^2)\, s^-(x_b, \theta_b^1)$ evaluates to 1 and equation (1) becomes $\dot{x}_a = \kappa_a - \gamma_a\, x_a$. From the inequalities (3), it follows that $x_a < \theta_a^1 < \kappa_a/\gamma_a$, so $\dot{x}_a > 0$. To each flow domain D corresponds a *qualitative state* QS defined as the tuple $\langle D, \boldsymbol{S} \rangle$, where the vector $\boldsymbol{S}$ represents the derivative sign of solutions in D. A qualitative state $QS = \langle D, \boldsymbol{S} \rangle$ is called *instantaneous*, if all solutions traverse D instantaneously and *persistent* otherwise. There is a *transition* from $QS^1 = \langle D^1, \boldsymbol{S}^1 \rangle$ to $QS^2 = \langle D^2, \boldsymbol{S}^2 \rangle$, if there exists a solution reaching D^2 from D^1. The set of qualitative states and transitions between qualitative states together form the *state transition graph*. The state transition graph corresponding to our two-gene example, represented in figure 2(b), contains 18 persistent qualitative states, including two stable (QS^6, QS^{22}) and one unstable (QS^{12}) qualitative equilibrium states.

So, using the simulation method sketched above, the qualitative behavior emerging from genetic regulatory interactions can be predicted. These results are obtained using a version of the GNA tool still under development [6]. The publicly-available version of GNA (GNA 5.0) gives similar results, but uses a slightly coarser partition of the phase space into domains, which makes the interpretation of the properties associated to qualitative states less straightforward.



(a) (b)

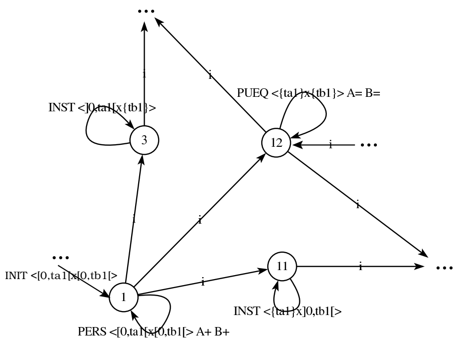
Fig. 2. (a) Partition of the phase space into 33 flow domains. Flow domains may be of dimension 2 (*e.g.*, D^1), 1 (*e.g.*, D^{11}) or 0 (*e.g.*, D^{12}). (b) State transition graph, corresponding to the two-gene example in figure 1(a). Filled and unfilled dots correspond to instantaneous and persistent qualitative states, respectively. Qualitative equilibrium states are circled in addition.

3 Transformation of Simulation Results into LTSs

First of all, it is necessary to translate the simulation result into a format suitable for verification. For this purpose, we developed a translator which takes as input a state transition graph produced by qualitative simulation and produces a corresponding LTS encoded in the BCG (*Binary Coded Graph*) format used by CADP [8].

To each qualitative state corresponds a state with a self-transition (loop) in the LTS. The label of this loop encodes all the properties of the corresponding qualitative state: its name, the range and derivative sign of protein concentrations, and additional properties specifying whether the state is instantaneous, persistent, or a stable or unstable equilibrium. Each transition between qualitative states is encoded in the LTS by an invisible transition (labeled by the action "`i`", noted τ in CCS). Since state transition graphs produced by qualitative simulation of genetic regulatory networks may be disconnected, we create an initial state which is linked to all other states in the LTS via special transitions.

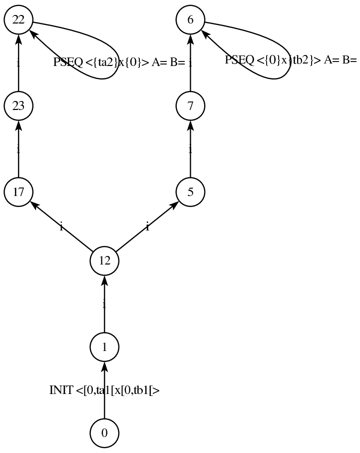
Figure 3(a) shows the translation of the persistent qualitative state QS^1 associated to the flow domain D^1. This qualitative state is encoded in the Lts by a state with a loop labeled `"PERS <[0,ta1[x[0,tb1[> A+ B+`". Three invisible transitions originate from this state, linking it to the states corresponding to QS^3, QS^{11}, and QS^{12}.



(a)

```
[ "INIT <[0,ta1[x[0,tb1[>" ]
 < true* . "PSEQ <{ta2}x{0}> A= B=" > true
 and
 < true* . "PSEQ <{0}x{tb2}> A= B=" > true
)
```

(b)



(c)

Fig. 3. (a) Fragment of the Lts corresponding to the qualitative state QS^1 and its successors. "INST", "PERS", "PSEQ" and "PUEQ" denote qualitative states that are instantaneous, persistent non equilibrium, persistent stable equilibrium, and persistent unstable equilibrium, respectively. The derivative sign of each protein concentration is represented by '=', '-' and '+'. (b) Bistability property formulated in regular alternation-free μ-calculus. (c) Diagnostic produced by model-checking.

A simplified Lts can be obtained from the original Lts by abstracting away the states corresponding to instantaneous qualitative states. By exploiting the fact that in the qualitative model there are never two successive instantaneous states, we can perform this simplification by hiding all labels corresponding to instantaneous qualitative states in the original Lts and minimizing it modulo branching bisimulation using the Bcg_Min tool of Cadp. The dynamical properties of interest are preserved in the simplified Lts.

4 Verification using Temporal Logic

Once the Lts corresponding to the genetic regulatory network has been generated by using Gna together with the translator described in Section 3, we can use the model checking technologies available in Cadp to analyze the behaviour of the biological system. The methodology adopted consists of two steps:

– First, each desired property is expressed as a formula in regular alternation-free μ-calculus [12], which is the input language of the EVALUATOR 3.0 model checker of CADP. This temporal logic is a good compromise between expressive power (it subsumes CTL and PDL), user-friendliness (concise formula descriptions due to regular expressions), and model-checking efficiency (algorithms linear w.r.t. formula size and LTS size). Also, generic properties can be encoded as macro definitions and grouped into reusable libraries.
– Second, each property is verified on the LTS using EVALUATOR 3.0, which produces diagnostics (counterexamples and witnesses) illustrating its truth value. The diagnostics obtained, represented as LTSs, can then be inspected visually using the BCG_EDIT graphical LTS editor of CADP. Diagnostics can also be replayed interactively in the LTS by means of the graphical simulator OCIS of CADP.

Figure 3 illustrates the verification of the *bistability property* on the LTS constructed from the genetic regulatory network given in Figure 1. This property states that from an initial state QS^1 in which both proteins A and B have low concentrations (below θ_a^1 and θ_b^1), it is possible to reach two different stable equilibrium states (QS^{22} and QS^6) in which only one protein is present and at a high concentration (at θ_a^2 and at θ_b^2, respectively). The diagnostic (witness) exhibited by EVALUATOR 3.0 is a LTS subgraph containing the paths going from the initial state to the two stable equilibrium states.

Other biologically interesting properties (*e.g.*, reachability of certain equilibrium states, existence of behaviours satisfying certain constraints on protein concentrations) can be verified in a similar way.

5 Conclusion

Our approach for analyzing biological systems consists in connecting GNA, a qualitative simulation tool well-adapted to the available information on genetic regulatory networks, to the widely-used CADP verification toolbox. By translating the state transition graph produced by GNA into a LTS, standard verification technologies become available for analyzing the dynamics of the underlying genetic regulatory network.

Checking properties of qualitative simulation results using temporal logic was originally proposed by Shults and Kuipers [14]. Chabrier and Fages [4] and Peres and Comet [13] have also addressed the formal analysis of genetic regulatory networks using model-checking approaches, but they use rather simple rule-based and Boolean models, respectively. Like us, Alur *et al.* [1], Antoniotti *et al.* [2] and Ghosh *et al.* [9] use hybrid models to analyze biological networks. However, Alur *et al.* and Antoniotti *et al.* use numerical instead of qualitative models. The most closely-related work is the symbolic reachability analysis of Ghosh *et al.*, but the authors ignore the problems related to discontinuities in the right-hand side of the differential equations. Additionally, GNA has been tailored so as to exploit the favorable mathematical properties of the piecewise-linear models, which may make it better capable of analyzing large and complex genetic regulatory networks. In previous work, we used CTL [3], as in [4] and [13], but the use of the more expressive and convenient regular alternation-free μ-calculus, together with the diagnostic generation and interactive simulation facilities offered by CADP, makes it possible to easily express properties, interpret the results of the analysis, and relate them to biological reality.

References

1. R. Alur, C. Belta, F. Ivančíc, V. Kumar, M. Mintz, G.J. Pappas, H. Rubin, and J. Schlug. Hybrid modeling and simulation of biomolecular networks. In M.D. Di Benedetto and A. Sangiovanni-Vincentelli, eds, *Hybrid Systems: Computation and Control (HSCC 2001)*, vol. 2034 of *LNCS*, 19–32. Springer, 2001.
2. M. Antoniotti, B. Mishra, C. Piazza, A. Policriti, and M. Simeoni. Modeling cellular behavior with hybrid automata: bisimulation and collapsing. In C.Priami, ed., *First Int. Work. Comput. Meth. Systems Biol. (CMSB 2003)*, vol. 2602 of *LNCS*, 57–74. Springer, 2003.
3. G. Batt, H. de Jong, J. Geiselmann, and M. Page. Analysis of genetic regulatory networks: A model-checking approach. In P. Salles and B. Bredeweg, eds, *Proc. 17th Int. Work. Qualitative Reasoning (QR-03)*, 31–38, Brasilia, Brazil, 2003.
4. N. Chabrier and F. Fages. Symbolic model checking of biochemical networks. In C.Priami, ed., *First Int. Work. Comput. Meth. Systems Biol. (CMSB 2003)*, vol. 2602 of *LNCS*, 149–162. Springer, 2003.
5. H. de Jong, J. Geiselmann, G. Batt, C. Hernandez, and M. Page. Qualitative simulation of the initiation of sporulation in *B. subtilis*. *Bull. Math. Biol.*, 2004. In press.
6. H. de Jong, J. Geiselmann, C. Hernandez, and M. Page. Genetic Network Analyzer: Qualitative simulation of genetic regulatory networks. *Bioinformatics*, 19(3):336–344, 2003. `http://www-helix.inrialpes.fr/gna`.
7. H. de Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bull. Math. Biol.*, 2004. In press.
8. H. Garavel, F. Lang, and R. Mateescu. An overview of CADP 2001. *Europ. Assoc. for Soft. Sci. and Tech. (EASST) Newsletter*, 4:13–24, 2002. Also INRIA Technical Report RT-0254 (2001). `http://www.inrialpes.fr/vasy/cadp`.
9. R. Ghosh, A. Tiwari, and C. Tomlin. Automated symbolic reachability analysis; with application to delta-notch signaling automata. In O. Maler and A. Pnueli, eds, *Hybrid Systems: Computation and Control (HSCC 2003)*, vol. 2623 of *LNCS*, 233–248, Springer, 2003.
10. L. Glass and S.A. Kauffman. The logical analysis of continuous non-linear biochemical control networks. *J. Theor. Biol.*, 39:103–129, 1973.
11. K.W. Kohn. Molecular interaction maps as information organizers and simulation guides. *Chaos*, 11(1):1–14, 2001.
12. R. Mateescu and M. Sighireanu. Efficient on-the-fly model-checking for regular alternation-free mu-calculus. *Sci. Comput. Program.*, 46(3):255–281, 2003.
13. S. Peres and J. P. Comet. Contribution of computational tree logic to biological regulatory networks: Example from *P. aeruginosa*. In C. Priami, ed., *First Int. Work. Comput. Meth. Systems Biol. (CMSB 2003)*, vol. 2602 of *LNCS*, 47–56, Springer, 2003.
14. B. Shults and B. J. Kuipers. Proving properties of continuous systems: Qualitative simulation and temporal logic. *Artif. Intell.*, 92(1-2):91–130, 1997.