Hybrid Automata in Systems Biology: How far can we go?

Hybrid Automata in System Biology: How far can we go?

Dario Campagna [1,2] and Carla Piazza [1,3]

Dept. of Mathematics and Computer Science
University of Udine
Via delle Scienze 206, 33100 Udine, Italy

Abstract

We consider the reachability problem on semi-algebraic hybrid automata. In particular, we deal with the effective cost that has to be afforded to solve reachability through first-order satisfiability.
The analysis we perform with some existing tools shows that even simple examples cannot be efficiently solved. We need approximations to reduce the number of variables in our formulae: this is the main source of time computation growth. We study standard approximation methods based on Taylor polynomials and ad-hoc strategies to solve the problem and we show their effectiveness on the repressilator case study.

Keywords: Hybrid Automata, Reachability, Semi-Algebraic First-Order Formulæ.

Introduction

Since their introduction (see, e.g., [5]), hybrid automata have initiated a new tradition, promising powerful tools for modeling and reasoning about complex engineered or natural systems (see, e.g., [1,21]).

Intuitively, a hybrid automaton consists of a finite graph, whose nodes are called *locations*, together with a set of continuous variables which evolve according to continuous laws, called *dynamics*, characterising each discrete location. The continuous evolution of the hybrid automaton may change from location to location. Moreover, each location is characterised by an *invariant* condition which defines the allowed values for the continuous variables inside the location. Finally, each graph's *edge* is labelled by both an *activation* condition and a *reset* map. The edge can be crossed only if the continuous variables satisfy the activation condition and after crossing it the continuous variables are set accordingly to the reset map. The double nature, both discrete and continuous, of hybrid automata make them particularly

[1] This work is partially supported by MIUR.

[2] Email: campagnadario@alice.it

[3] Email: carla.piazza@dimi.uniud.it

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

suitable in the modeling of systems exhibiting a mixed behaviour which cannot be characterised in a proper way using either discrete or continuous formalisms.

In this context, one of the basic problems is the *reachability* one which requires to decide whether it is possible to move from a state (a pair consisting of a location together with a set of values for the continuous variables) to another.

Unfortunately, the flexibility and expressive power of hybrid automata soon lead to undecidability and complexity results [25] which cast doubts on their suitability as a general tool that can be algorithmized and efficiently implemented.

In order to control both undecidability and complexity one can either impose syntactic conditions and concentrate on classes of hybrid automata or define semantic approximation techniques.

In [31] the class of *semi-algebraic hybrid automata* has been introduced. The invariants, dynamics, activations, and resets of semi-algebraic automata have to be first-order formulæ over the theory of $(\mathbb{R}, 0, 1, +, *, <)$. On the one hand, such formulæ are decidable [37] and tools such as QEPCAD B [13] can be used to manage them. On the other hand, Taylor polynomials allow to use semi-algebraic formulæ to approximate with arbitrary precision any smooth function. As a consequence of the expressive power of semi-algebraic hybrid automata, the undecidability of the reachability problem for such class can be proved [30]. In particular, in this case, undecidability is a consequence of the fact that we cannot a-priori bound the number of edges we need to cross. Hence, we can see "the glass half full" saying that *bounded* (w.r.t. edge crossing) reachability is computable. Unfortunately, as noticed in [29] such computation results to be too time/space consuming due to the high computational complexity of semi-algebraic decomposition.

In this paper we start from the considerations presented in [29] concerning the effectiveness of bounded reachability computation on semi-algebraic hybrid automata and we show on some examples which kind of approximations are necessary to keep complexity under control. As done in [29] we may distinguish space and time discretizations in our work. As far as space discretizations are concerned, instead of implementing an ad-hoc algorithm, we try to exploit tools which allow approximate computations over the reals such as RSOLVER [33] and ECL^{i}PS^e [7]. Unfortunately, this is not enough: space approximations which *separate* the continuous variables are necessary. We notice that time discretization and Taylor polynomials are essential ingredients in our approach.

The paper is organized as follows. In Section 1 we quickly overview the state of the art. Some basic notions about semi-algebraic hybrid automata and reachability find place in Section 2, while Section 3 is the core part of our work. In Section 4 we apply our analysis to the Repressilator case study. Some conclusions are drawn in Section 5.

1 Related Works

As mentioned in the introduction, we can control undecidability and complexity on hybrid automata in two ways: imposing syntactic constraints which limit the expressive power or introducing semantic approximation techniques.

In [2] Alur et al. introduced *multirate automata* as an extensions of *timed au-*

tomata [4]. Such hybrid automata are characterised by resets which are either identity or constant function zero. Moreover, their continuous variables evolve like clocks with rational rates. In the same work it has been proved that the reachability problem over multirate automata is not decidable in general. However, imposing a restriction on dynamics called *simplicity condition*, decidability for reachability problem and finite bisimulation are proved. Puri and Varaiya in [32] introduced *rectangular hybrid automata* whose dynamics can be characterised by a differential inclusion. They showed that, under a condition called *initialized condition*, reachability can be decided. Lafferriere, Pappas and Sastry introduced *o-minimal hybrid automata* in [27]. Such class of hybrid automata guarantee finite bisimulation quotient imposing both constant reset condition to all the edges and a unique o-minimal dynamic from each state. In [14] it has been proved that reachability is still decidable on semi-algebraic o-minimal automata when the conditions on the dynamics are relaxed allowing many possible continuous evolutions. Unfortunately, all the above mentioned classes have restrictions on both dynamics and resets and thus they are not suitable to verify properties of many interesting hybrid systems.

As far as approximation techniques are concerned, in [23] Halbwachs et al. suggested convex approximations as a way to verify linear hybrid systems, Dang and Maler proposed to verify hybrid automaton properties via face lifting in [17], Chutinan and Krogh showed in [15] how evolutions of polyhedral-invariant hybrid automata can be approximated using polyhedra, Asarin et al. gave in [9] a technique to approximate reachability analysis of piecewise-linear dynamical systems, Kurzhanski and Varaiya introduced ellipsoidal techniques in [26], Alur et al. proposed in [3] *predicate abstraction* as a technique to perform reachability analysis. Many tools, based on such techniques, have been developed in the last years. In particular, we can recall *HyTech* [24], *d/dt* [8], *Checkmate* [35], *UPPAAL* [11], and *KRONOS* [18]. Unfortunately, all these approximation methods and tools are again defined on restricted classes of hybrid automata. Such classes are clearly larger than the classes on which decidability has been proved. However, it is still necessary to check that the model satisfies all the required conditions before the method can be applied.

Semi-algebraic hybrid automata introduced in [31] intrinsically combine syntactic restrictions and semantics approximations. On the one hand Taylor polynomials can be used to approximate a large class of hybrid automata with semi-algebraic ones. In [28] Lanotte and Tini proposed an approximation technique for hybrid automata that exploits Taylor polynomials to obtain from an hybrid automaton H a polynomial hybrid automaton H' that over-approximate H. On the other hand, cylindrical algebraic decomposition (CAD) algorithms (see, e.g., [16,22,34,10]) can be used to reason on semi-algebraic hybrid automata. Such considerations are also at the basis of the abstractions and analysis techniques presented in [21].

The tool QEPCAD B [13] efficiently implements Collins' CAD-based algorithm [16] for quantifier elimination, transforming any given first-order semi-algebraic formula into an equivalent quantifier-free one and it can easily become the engine of a step-by-step reachability algorithm for semi-algebraic automata. Unfortunately, the computational cost is still too high. QEPCAD B is not the only tool which can be used to manage constraints over the reals. In particular, we recall: RSOLVER [33], a program for solving quantified inequality constraints over the reals based on

a *branch-and-prune* algorithm; ECL^{i}PS^e [7], a software system for the development and deployment of constraint programming applications that contains a general interval propagation solver which can be used to solve problems over both integer and real variables; Redlog [19], a package that extends the computer algebra system reduce to a system that provides algorithms for the symbolic manipulation of first-order formulæ with some syntactic restrictions on the quantified variables; clp(rl) [36], a constraint solving system, implemented on top the computer logic system Redlog, where the admissible constraints are arbitrary first-order formulæ.

2 Reachability in Semi-Algebraic Hybrid Automata

In this section we introduce the standard syntax and semantics of hybrid automata and describe the reachability problem on semi-algebraic hybrid automata.

We start with some notations and conventions we use on hybrid automata. Capital letters Z_1, Z_2,..., Z_m, Z'_1,..., Z'_m,..., denote variables ranging over $\mathbb{R}$. Analogously, Z denotes the vector of variables $\langle Z_1, \ldots, Z_d\rangle$ and Z' denotes the vector $\langle Z'_1, \ldots, Z'_d\rangle$. The temporal variables $T, T', T'', \ldots$ model time and range over $\mathbb{R}_{\geq 0}$. We use the small letters p, q, r, s, ... to denote d-dimensional vectors of real numbers. Occasionally, we may use the notation $\varphi[X_1, \ldots, X_m]$ to stress the fact that the set of free variables of the first-order formula φ is included in the set of variables $\{X_1, \ldots, X_m\}$. By extension, if $\{Z_1, \ldots, Z_n\}$ is a set of variable vectors, $\varphi[Z_1, \ldots, Z_n]$ indicates that the free variables of φ are included in the set of components of $Z_1, \ldots, Z_n$. Moreover, given a formula $\varphi[Z_1, \ldots, Z_i, \ldots, Z_n]$ and a vector p of the same dimension as the variable vector Z_i, the formula obtained by component-wise substitution of Z_i with p is denoted by $\varphi[Z_1, \ldots, Z_{i-1}, p, Z_{i+1}, \ldots, Z_n]$. When in φ the only free variables are the components of Z_i, after the substitution we can determine the truth value of $\varphi[p]$.

Hybrid automata have a mixed discrete and continuous behaviour. The discrete component is represented by a graph, while the continuous one is given as a set of continuous variables. For each node of the discrete graph we have an invariant condition and a dynamic law over the continuous variables. The dynamic law may depend on the initial conditions, i.e., on the values of the continuous variables at the beginning of the evolution in the state. The jumps from one discrete state to another are regulated by activation and reset conditions on the continuous variables.

Definition 2.1 [Hybrid Automata - Syntax] A *hybrid automaton* $H = (Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Res)$ of dimension d consists of the following components:

(i) $Z = \langle Z_1, \ldots, Z_d\rangle$ and $Z' = \langle Z_1', \ldots, Z_d'\rangle$ are two vectors of variables ranging over the reals $\mathbb{R}$;

(ii) $\langle \mathcal{V}, \mathcal{E}\rangle$ is a graph. Each element of $\mathcal{V}$ will be dubbed *location*.

(iii) Each vertex $v \in \mathcal{V}$ is labeled by the formulæ $Inv(v)[Z]$ and $Dyn(v)[Z, Z', T] \equiv Z' = f_v(Z, T)$, where $f_v : \mathbb{R}^d \times \mathbb{R}_{\geq 0} \longrightarrow \mathbb{R}^d$;

(iv) Each edge $e \in \mathcal{E}$ is labeled by the two formulæ $Act(e)[Z]$ and $Res(e)[Z, Z']$.

The semantics of hybrid automata regulates the time evolution of the continuous variables.

Definition 2.2 [Hybrid Automata - Semantics] A *state* ℓ of H is a pair $\langle v, r\rangle$, where $v \in \mathcal{V}$ is a location and $r = \langle r_1, \ldots, r_d\rangle \in \mathbb{R}^{d(H)}$ is an assignment of values for the variables of Z. A state $\langle v, r\rangle$ is said to be *admissible* if $Inv(v)[r]$ is true.

The *continuous reachability transition relation* $\xrightarrow{t}_C$, with $t > 0$ is the transition elapsed time, between admissible states is defined as follows:
$\langle v, r\rangle \xrightarrow{t}_C \langle v, s\rangle$ iff it holds that $s = f_v(r, t)$, and for each $t' \in [0, t]$ the formula $Inv(v)[f_v(r, t')]$ is true.

The *discrete reachability transition relation* $\xrightarrow{e}_D$ between admissible states is defined as follows:
$\langle v, r\rangle \xrightarrow{e}_D \langle u, s\rangle$ iff both $Act(e)[r]$ and $Res(e)[r, s]$ are true.

We use the notation $\ell \to \ell'$ to denote that either $\ell \xrightarrow{t}_C \ell'$ or $\ell \xrightarrow{e}_D \ell'$, for some $t \in \mathbb{R}_{\geq 0}$, $e \in \mathcal{E}$.

A trace is a sequence of continuous and discrete transitions. A point s is reachable from a point r if there is a trace starting from r and ending in s.

Definition 2.3 [Hybrid Automata - Reachability] A *trace* of H is a sequence of admissible states $[\ell_0, \ell_1, \ldots, \ell_i, \ldots, \ell_n]$ such that $\ell_{i-1} \to \ell_i$ holds for each $1 \leq i \leq n$.

The automaton H *reaches* a point $s \in \mathbb{R}^d$ (in time t) from a point $r \in \mathbb{R}^d$ if there exists a trace $tr = [\ell_0, \ldots, \ell_n]$ of H such that $\ell_0 = \langle v, r\rangle$ and $\ell_n = \langle u, s\rangle$, for some $v, u \in \mathcal{V}$ (and t is the sum of the continuous transitions elapsed times). In such a case, we also say that s is *reachable* from r in H.

A path ph over a graph G is a sequence $[v_0, \ldots, v_n]$ of nodes of G such that for each $1 \leq i \leq n$ there is an edge from v_{i-1} to v_i. Given a hybrid automaton H and trace, tr, of H, a *corresponding path* of tr is a path ph obtained by considering the discrete transitions occurring in tr.

We are interested in the reachability problem for hybrid automata, namely, given a hybrid automaton H, an initial set of points $I \subseteq \mathbb{R}^d$, and a final set of points $F \subseteq \mathbb{R}^d$ we wish to decide whether there exists a point in I from which a point in F is reachable.

An interesting class of hybrid automata is the class of *semi-algebraic hybrid automata* [31].

Definition 2.4 [Semi-Algebraic Automata] A hybrid automaton is *semi-algebraic* if $Dyn(v)$, $Inv(v)$, $Act(e)$, and $Res(e)$ are formulæ belonging to the first-order theory of $(\mathbb{R}, 0, 1, +, *, <)$ [37], also known as the theory of semi-algebraic sets.

Moreover, we say that H is *continuous* if $\forall v \in \mathcal{V}$ $f_v(Z, T)$ is continuous on $\mathbb{R}^d \times \mathbb{R}_{\geq 0}$ and $f_v(r, 0) = r$, for each $r \in \mathbb{R}^d$.

In the rest of this paper we concentrate on continuous semi-algebraic hybrid automata, avoiding all the technical problems concerning the existence, uniqueness and continuity of dynamics (see [14] for more details).

The reachability problem for such class of automata is semi-decidable and it can be reduced to the satisfiability of a numerable disjunction of formulæ of the form $Reach(ph)[Z, Z']$ [14]. In particular, if H is a semi-algebraic automaton, then $q \in \mathbb{R}^d$ is reachable from $p \in \mathbb{R}^d$ in H through a trace whose corresponding path is ph if and only if the formula $Reach(ph)[p, q]$ holds. Unfortunately, as proved in [30],

the reachability problem for semi-algebraic automata remains undecidable even if we consider computational models over the reals.

Now let us have a closer look at the first-order formulæ involved in the reachability computation. Inside a discrete location v the following formula expresses that Z reaches Z':

$$Reach(v)[Z, Z'] \equiv Inv(v)[Z] \wedge \exists T \geq 0(Z' = f_v(Z, T) \wedge \forall 0 \leq T' \leq T(Inv(v)[f_v(Z, T')]))$$

On the other hand, when we cross an edge $\langle v, u\rangle$ we have to consider the formula:

$$Reach(\langle v, u\rangle)[Z, Z'] \equiv Inv(v)[Z] \wedge Act(\langle v, u\rangle)[Z] \wedge Res(\langle v, u\rangle)[Z, Z'] \wedge Inv(u)[Z']$$

Combining the above formulæ, for each path ph we can easily construct the formula $Reach(ph)[Z, Z']$. For instance if we have the path $ph = [v, u]$, then:

$$Reach([v, u])[Z, Z'] \equiv \exists Z'', Z'''(Reach(v)[Z, Z''] \wedge Reach(\langle v, u\rangle)[Z'', Z'''] \wedge Reach(u)[Z''', Z'])$$

Example 2.5 Let $H_1 = (Z,\ Z',\ \mathcal{V},\ \mathcal{E},\ Inv,\ Dyn,\ Act,\ Res)$ where:

- Z, Z' are variables over $\mathbb{R}$,
- $\mathcal{V} = \{v, u\}$ and $\mathcal{E} = \{e\}$, where e goes from v to u,
- $Inv(v)[Z] \equiv 1 \leq Z \leq 10$ and $Inv(u)[Z] \equiv 10 \leq Z \leq 20$,
- $Dyn(v)[Z, Z', T] \equiv Z' = Z + (2Z^2 + Z)T$ and $Dyn(u)[Z, Z', T] \equiv Z' = Z + (3Z^2 + Z)T$,
- $Act(e)[Z] \equiv Z = 10$,
- $Res(e)[Z, Z'] \equiv Z' = Z$.

The formula for the path $ph = [v, u]$ is the following:

$$\begin{aligned} Reach([v, u])[Z, Z'] \equiv \exists Z'', Z'''\Big(&Inv(v)[Z] \wedge \exists T \geq 0\big(Z'' = Z + (2Z^2 + Z)T \wedge \\ &\forall 0 \leq T' \leq T(Inv(v)[Z + (2Z^2 + Z)T'])\big) \wedge \\ &Inv(v)[Z''] \wedge Act(e)[Z''] \wedge Res(e)[Z'', Z'''] \wedge Inv(u)[Z'''] \wedge \\ &\exists T'' \geq 0\big(Z' = Z''' + (3Z'''^2 + Z''')T'' \wedge \\ &\forall 0 \leq T''' \leq T''(Inv(u)[Z''' + (3Z'''^2 + Z''')T'''])\big)\Big) \end{aligned}$$

3 Solving the Reachability Problem

In this section we describe some approximation methods for the reachability problem on semi-algebraic hybrid automata. All the computations have been performed on a Dual Core AMD Opteron^{TM} Processor 275, 2205.042 MHz with 4 GB RAM, running CentOS.

The complexity of the reachability formulæ presented in Section 2 increases with the length of the discrete path. In particular, we can notice that the degree of the involved polynomials and the quantifier alternation remains bounded, while the number of variables linearly increases.

Since the first-order theory of $(\mathbb{R}, 0, 1, +, *, <)$ admits the quantifier elimination, we can try to bound the number of variables in each formulæ. When we apply the quantifier elimination procedure to $Reach(ph)[Z, Z']$ we obtain an equivalent first-order formula $\phi[Z, Z']$ involving only the variables Z and Z'. If we now add a step to the path $ph = [v_1, \ldots, v_n]$, i.e., we consider the path $ph' = [v_1, \ldots, v_n, v_{n+1}]$, we only have to apply quantifier elimination to the formula:

$$\exists Z'', Z'''(\phi[Z, Z''] \wedge Reach(\langle v_n, v_{n+1}\rangle)[Z'', Z'''] \wedge Reach(v_{n+1})[Z''', Z'])$$

Proceeding in this way, it seems that we can keep under control the complexity of our method. Unfortunately, if we try to apply it, exploiting Qepcad b to obtain quantifier free formulæ at each step, we cannot go far enough, as shown by the following example.

Example 3.1 Consider the following hybrid automaton.
$H_2 = (Z,\ Z',\ \mathcal{V},\ \mathcal{E},\ Inv,\ Dyn,\ Act,\ Res)$ where:

- $Z = \langle Z_1, Z_2\rangle$ and $Z' = \langle {Z_1}', {Z_2}'\rangle$, where Z_1, Z_2, ${Z_1}'$, ${Z_2}'$ variables over $\mathbb{R}$,
- $\mathcal{V} = \{v, u\}$ and $\mathcal{E} = \{e\}$, where e goes from v to u,
- $Inv(v)[Z] \equiv 1 \leq Z_1 \leq 10 \wedge 1 \leq Z_2 \leq 10$ and
  $Inv(u)[Z] \equiv 10 \leq Z_1 \leq 20 \wedge 10 \leq Z_2 \leq 20$,
- $Dyn(v)[Z, Z', T] \equiv {Z_1}' = Z_1 + (2Z_1^2 + Z_1)T \wedge {Z_2}' = Z_2 + (2Z_2^2 + Z_2)T$ and
  $Dyn(u)[Z, Z', T] \equiv {Z_1}' = Z_1 + (3Z_1^2 + Z_1)T \wedge {Z_2}' = Z_2 + (3Z_2^2 + Z_2)T$,
- $Act(e)[Z] \equiv Z_1 = 10 \wedge Z_2 = 10$,
- $Res(e)[Z, Z'] \equiv {Z_1}' = Z_1 \wedge {Z_2}' = Z_2$.

Suppose we want to apply the method described above with $ph = [v, u]$. First, we use Qepcad b to compute a quantifier free formula $\phi[Z, Z']$ equivalent to the formula $Reach(v)[Z, Z']$. Then we construct the formula:

$$\exists Z'', Z'''(\phi[Z, Z''] \wedge Reach(\langle v, u\rangle)[Z'', Z'''] \wedge Reach(u)[Z''', Z'])$$

When we try to compute an equivalent quantifier free formula with Qepcad b we find out that we cannot obtain any result within 20 minutes of CPU time.

Using this method we are able to limit the number of variables in our formulæ, but we have an increasing number of polynomials and constraints in the computed quantifier free formulæ. This is one of the problems of this method, since the complexity of the new constructed formulæ strongly depends on the number of polynomials and constraints occurring in computed quantifier free formulæ. Another problem of the method is that Qepcad b could not give any result in reasonable time when used on formulæ of the form $Reach(v)[Z, Z']$, i.e., the reachability problem inside a location could be already too complex.

At this point the only possibility we have is that of introducing approximations.

A first approximated approach to the reachability problem consists in the application of the above method exploiting RSolver instead of Qepcad b. Acting in this way we hope to solve both the problems mentioned in Example 3.1. Unfortunately, this approach is less effective than the previous one.

Example 3.2 Consider the hybrid automaton H_2 of example 3.1. RSolver on the formula $Reach(v)[Z, Z']$ gives the following result:

```
True, volume ~[ 0., 0.]
False, volume ~[ 5905.08179397, 5905.08179397]
⋮
Unknown:
⋮
```

Since the `True` set is empty we do not know which values of Z and Z' satisfy the formula $Reach(v)[Z, Z']$ and we cannot procede with the next step.

The results obtained with RSolver on formulæ of the form $Reach(v)[Z, Z']$ are too approximated for being used. However, we can use it to try to solve the problem related to the number of polynomials and constraints appearing in computed quantifier free formulæ. To do this we apply the previous method exploiting Qepcad b with the add of an intermediate step that involves the use of RSolver.

More precisely, consider the path $ph' = [v_1, \ldots, v_n, v_{n+1}]$ and suppose we have already computed a quantifier free formula $\phi[Z, Z']$ equivalent to $Reach(ph)[Z, Z']$, where $ph = [v_1, \ldots, v_n]$. Using RSolver we compute an approximation of the set of values for Z and Z' that satisfy $\phi[Z, Z']$, then we construct a first-order formula $\gamma[Z, Z']$ defining such approximation. Finally, we apply the quantifier elimination procedure to the formula:

$$\exists Z'', Z'''(\gamma[Z, Z''] \wedge Reach(\langle v_n, v_{n+1}\rangle)[Z'', Z'''] \wedge Reach(v_{n+1})[Z''', Z'])$$

It is still not enough, as shown by the following example.

Example 3.3 Consider again the hybrid automaton H_2 of Example 3.1. Let $\phi[Z, Z']$ be the quantifier free formula equivalent to $Reach(v)[Z, Z']$ computed by Qepcad b. RSolver on the formula $\phi[Z, Z']$ gives the following result:

```
True, volume ~[ 0., 0.]
False, volume ~[ 5904.9114008, 5904.91140081]
⋮
Unknown:
⋮
```

As in Example 3.2 we obtain an empty `True` set and we cannot procede with the next step.

Another approximated approach that we can consider consists in the application of this last described method using ECL^{i}PS^e instead of RSolver to compute the set of values that satisfy a quantifier free formula obtained with Qepcad b.

Given a quantifier free formula we can define a constraint satisfaction problem with constraint on reals that can be solved by ECL^{i}PS^e through constraint propagation and search techniques. An answer to a problem on reals is called conditional

solution. The number of conditional solutions returned vary according to the level of precision in the search procedure. For instance, given the problem defined from the formula $\phi[Z, Z']$ of Example 3.3 and using the predicate `locate/2` with final precision 1.0 ECL^{i}PS^e returns 51 answers, if we reduce the final precision to 0.1 we obtain more than 102 answers. Even if we find a way to use the values computed by ECL^{i}PS^e to construct the formula for the successive step, the method would not be effective, because we still have the problem that Qepcad b could not give any result when used on formulæ of the form $Reach(v)[Z, Z']$.

All the above discussed methods share one problem: the high cost in terms of computation time that has to be afforded to compute a quantifier free formula from a formula of the form $Reach(v)[Z, Z']$ using Qepcad b. We have to find an approximation strategy to solve this problem in order to obtain an effective method to compute approximated solutions for the reachability problem.

To achieve this goal we studied a method to over-approximated the set of values reachable inside a discrete location of an automaton with independent dynamics.

Definition 3.4 [Hybrid Automata with Independent Dynamics] Le H be a continuous semi-algebraic hybrid automaton, let $Z = \langle Z_1, \ldots, Z_d \rangle$ and $Z' = \langle Z_1{}', \ldots, Z_d{}' \rangle$. H has independent dynamics if $\forall v \in \mathcal{V}$ the formula $Dyn(v)[Z, Z', T]$ is of the form:

$$Z_1{}' = f_{v,1}(Z_1, T) \wedge Z_2{}' = f_{v,2}(Z_2, T) \wedge \ldots \wedge Z_d{}' = f_{v,d}(Z_d, T)$$

Example 3.5 The automaton H_2 of Example 3.1 is a continuous semi-algebraic hybrid automaton with independent dynamics.

Given a discrete location v of an automaton with independent dynamics, we over-approximate the set of values Z' that can be reached inside v after time δ from values Z satisfying $Inv(v)[Z]$ applying the quantifier elimination procedure to the following formula

$$ReachApprox(v)[Z'] \equiv \exists Z (Inv(v)[Z] \wedge Z' = f_v(Z, \delta) \wedge Inv(v)[Z'])$$

This is an over-approximation of the sets of points reachable at time δ, since we did not check that at each time T' between 0 and δ the invariant is satisfied by $f_v(Z, T')$. Notice also that we can replace the condition $Inv(v)[Z]$ with a stronger one if we are interested in a subset of starting points. Using this formula many times we can compute an over-approximation of all the values Z' that can be reached with δ-time steps from values Z satisfying $Inv(v)[Z]$. After each step of duration δ we can consider the following formula to check if the edge $\langle v, u \rangle$ can be crossed:

$$\begin{aligned} ReachApprox(\langle v, u \rangle)[Z'] \equiv \exists Z (\phi[Z] \wedge Act(\langle v, u \rangle)[Z] \wedge \\ Res(\langle v, u \rangle)[Z, Z'] \wedge Inv(u)[Z']) \end{aligned}$$

where $\phi[Z']$ is a quantifier free formula equivalent to $ReachApprox(v)[Z']$. We apply the quantifier elimination procedure to this formula. If it results to be false, we increase the value of δ to compute another quantifier free formula $\phi[Z']$. Otherwise, we obtain a quantifier free formula $\psi[Z']$ and we can move to the discrete location

u where we can apply this procedure considering the formula:

$$ReachApprox(u)[Z'] \equiv \exists Z(\psi[Z] \wedge Z' = f_u(Z, \delta) \wedge Inv(u)[Z'])$$

Proceeding in this way we can keep under control the complexity of our formulæ, avoiding increases in the number of variables and polynomials. Exploiting Qepcad b to apply this method on the automaton H_2 of Example 3.1 we can prove that the discrete location u can be reached from v. The result is obtained in 30 milliseconds. Using this method we are able to find approximated solutions for the reachability problem also on automata with independent dynamics with more continuous variables and more complex formulæ than the ones occurring in H_2.

The method can be applied also to automata with non-independent dynamics, but it does not help us, as shown by the following example.

Example 3.6 Consider the following hybrid automaton with non-independent dynamics. $H_3 = (Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Res)$ where:

- $Z = \langle Z_1, Z_2 \rangle$ and $Z' = \langle {Z_1}', {Z_2}' \rangle$, where Z_1, Z_2, ${Z_1}'$, ${Z_2}'$ variables over $\mathbb{R}$,
- $\mathcal{V} = \{v, u\}$ and $\mathcal{E} = \{e\}$, where e goes from v to u,
- $Inv(v)[Z] \equiv 1 \leq Z_1 \leq 10 \wedge 1 \leq Z_2 \leq 8$ and
  $Inv(u)[Z] \equiv 8 \leq Z_1 \leq 50 \wedge 7 \leq Z_2 \leq 30$,
- $Dyn(v)[Z, Z', T] \equiv {Z_1}' = Z_1 + (2Z_1^2 + Z_1Z_2)T \wedge {Z_2}' = Z_2 + (7Z_2^2 + Z_2Z_1)T$ and
  $Dyn(u)[Z, Z', T] \equiv {Z_1}' = Z_1 + (3Z_1^2 + Z_1Z_2)T \wedge {Z_2}' = Z_2 + (4Z_2^2 + Z_2Z_1)T$,
- $Act(e)[Z] \equiv Z_1 \geq 8 \wedge Z_2 \geq 7$,
- $Res(e)[Z, Z'] \equiv {Z_1}' = Z_1 \wedge {Z_2}' = Z_2$.

Suppose we want to apply the method to find out if the discrete location u is reachable from v. First, we have to apply the quantifier elimination procedure to the formula $ReachApprox(v)[Z']$ with a fixed value δ. When we try to do this using Qepcad b we are not able to obtain any result within 20 minutes of CPU time because of the presence of non-independent dynamics.

To find approximated solution for the reachability problem on automata with non-independent dynamics, we have to introduce further approximations in our last method. Let H be an automaton with non-independent dynamics. We have that $\forall v \in \mathcal{V}$ the formula $Dyn(v)[Z, Z', T]$ is of the form:

$${Z_1}' = f_{v,1}(Z, T) \wedge {Z_2}' = f_{v,2}(Z, T) \wedge \ldots \wedge {Z_d}' = f_{v,d}(Z, T)$$

Given the formula $ReachApprox(v)[Z']$ and $\delta > 0$, we compute $\forall = 1, \ldots, d$ the minimum value ($min_i(v)$) and the maximum value ($max_i(v)$) that the function $f_{v,i}(Z, \delta)$ assume in the set defined by the formula $Inv(v)[Z]$. In order to determine an approximation of the values Z' that can be reached after time δ from values Z satisfying $Inv(v)[Z]$, we evaluate the following formula:

$$\bigwedge_{i=1,\ldots,d} min_i(v) \leq {Z_i}' \leq max_i(v) \wedge Inv(v)[Z']$$

If, in the previous method, we use this procedure instead of the quantifier elimination procedure to obtain a formula $\phi[Z']$ from formula $ReachApprox(v)[Z']$, we have a new method that can find approximated solution to the reachability problem even in presence of non-independent dynamics.

Example 3.7 Consider the hybrid automaton of Example 3.6. Suppose we want to apply the approximated method described above exploiting Qepcad b to find out if the discrete location u can be reached from v. First, we compute a formula $\phi[Z']$ using the procedure based on the calculus of minimum and maximum of each function in $Dyn(v)[Z, Z', T]$ described above. Then we apply the quantifier elimination procedure to the formula:

$$\exists Z(\phi[Z] \wedge Act(\langle v, u\rangle)[Z] \wedge Res(\langle v, u\rangle)[Z, Z'] \wedge Inv(u)[Z'])$$

We obtain a quantifier free formula representing the values for ${Z_1}'$ and ${Z_2}'$ in the discrete location u that can be reached starting from v. We succeed in proving the desired property (result obtained in 55 milliseconds).

We notice that solutions computed with this method are neither over nor under approximations.

Exploiting this last method together with Qepcad b, we could not be able to obtain results on some automata. In particular, on automata whose dynamics are either very complex or not representable in Qepcad b, e.g., functions where non integer or negative exponents appear. We can solve this problem introducing a further approximation to our method.

Consider a formula $Dyn(v)[Z, Z', T] \equiv \bigwedge_{i=1,\ldots,d} Z_i' = f_{v,i}(Z, T)$. Instead of computing the maximum and the minimum of $f_{v,i}(Z, \delta)$ in the set defined by the formula $Inv(v)[Z]$, we can compute the maximum and the minimum of the linearization of $f_{v,i}$ that is the Taylor polynomial of degree one:

$$Z_i'(\delta) = f_{v,i}(Z(0), 0) + \frac{df_{v,i}}{dT}(Z(0), 0)\delta + R$$

where R is the reminder term. In order to compute the maximum and the minimum at time δ_j (where $\delta_0 = \delta$ and $\delta_j > \delta_{j-1}$) we consider the following expression:

$$Z'(\delta_j) = Z'(\delta_{j-1}) + \frac{df_{v,i}}{dT}(Z(0), \delta_{j-1})\delta_j + R$$

Notice that the derivative $df_{v,i}/dT$ has not to be computed for every time interval. Once computed we can obtain a function that can be used to calculate the values of the derivative for all the different δ_j.

4 The Repressilator Case Study

As a simple yet very interesting example, we consider the Repressilator system constructed by Elowitz and Leibler [20]. It consists of three proteins, namely lacI, tetR, and cI, and the corresponding genes. The protein lacI represses the gene which expresses tetR, tetR represses the gene which expresses cI, whereas cI represses the

gene which expresses lacI, thus completing a feedback system. The dynamics of the network depend on the transcription rates, translation rates, and decay rates. Depending on the values of these rates the system might converge to a stable limit circle or become unstable.

We apply our method to compare the behaviour of two oscillating models proposed for the Repressilator.

First, we consider the hybrid automaton proposed in [12] to model the Repressilator. This hybrid automaton has 8 discrete locations, corresponding to all the possible combinations of genes being either *on* or *off*, and 9 variables. Three of them, A, B, C, represent the quantity of proteins in the system, the other six, $Y_{X,on}$, $Y_{X,off}$, where $X \in \{A, B, C\}$, control activation and deactivation of genes. For each discrete location v, $Inv(v) \equiv true$.

The differential equations governing proteins concentrations in each discrete location are decoupled: for instance, when gene A is *on* its dynamics is $\dot{A} = k_p - k_d A$, where k_p and k_d are costant parameters of the system.

The interactions between repressors and genes are confined to the activation conditions of the automaton transitions. Consider again gene A and suppose to be in a discrete location of the automaton where it is *on*. Then, the differential equation for $\dot{Y}_{A,off}$ is $\dot{Y}_{A,off} = k_b C$, the transition switching this gene *off* has an activation condition equal to $Y_{A,off} \geq 1 \wedge C \geq 1$ and a reset condition equal to $Y_{A,on} = 0 \wedge Y_{A,off} = 0$. The transition that turns gene A *on*, instead, has a constant rate k_u, hence its activation condition is $Y_{A,on} \geq 1$, $\dot{Y}_{A,on} = k_u$ is the differential equation for $\dot{Y}_{A,on}$ and the reset condition is equal to $Y_{A,on} = 0 \wedge Y_{A,off} = 0$, where k_b and k_u are costant parameters of the system.

In order to obtain a continuous semi-algebraic automaton from this hybrid automaton, we have only to define for each discrete location v a formula $Dyn(v)$ satisfying the conditions of Definition 2.4. To this aim we approximate the solutions of the differential equations in each discrete location with the corresponding Taylor polynomial of degree two. Consider, for instance, the differential equations for A, $Y_{A,off}$, and $Y_{A,on}$ in a discrete location where gene A is *on*, we approximate their solution with the following polynomials:

$$
\begin{aligned}
A' &= A + (k_p - k_d A)T + (-k_d k_p + k_d^2 A)T^2/2 \\
Y'_{A,off} &= Y_{A,off} + (k_b C)T + (-k_b k_d C)T^2/2 && \text{if gene } C \text{ is } off \\
Y'_{A,off} &= Y_{A,off} + (k_b C)T + (k_b k_p - k_b k_d C)T^2/2 && \text{if gene } C \text{ is } on \\
Y'_{A,on} &= Y_{A,on} + k_u T
\end{aligned}
$$

The solution of the differential equation for A in a discrete location where the gene A is *off* is approximated with the following polynomial:

$$A' = A + (-k_d A)T + (k_d^2 A)T^2/2$$

The automaton we obtain has non-independent dynamics (see, e.g., the equation for $Y'_{A,off}$), hence we analyse it using the approximated method based on the computation of minimum and maximum values of the functions defining the dynamics.

Starting from the discrete location where only gene A is active and with fixed values for proteins concentrations we succeed in simulating the automaton and observe an oscillatory behaviour (Fig. 1).

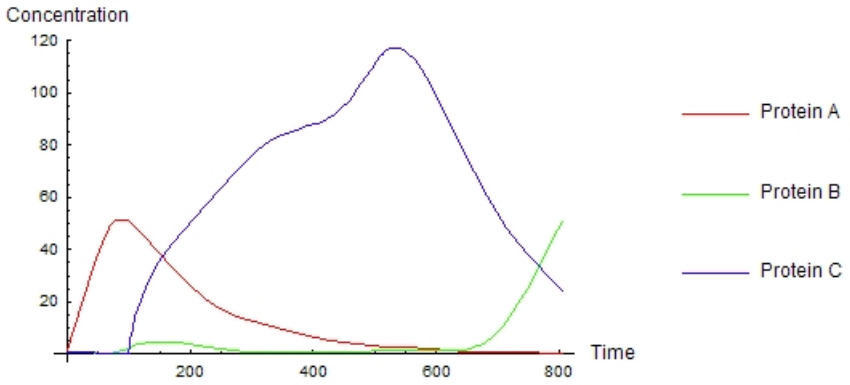

Fig. 1. Time trace of the hybrid automata with 8 discrete locations. Parameters are $k_p = 1$, $k_d = 0.01$, $k_b = 1$, $k_u = 0.01$.

The second hybrid automata we consider is the one that can be constructed from the following model for the repressilator written in the S-System equations formalism [38] (see [6])

$$\begin{aligned}
\dot{X}_1 &= \alpha_1 X_3^{-1} - \beta_1 X_1^{0.5}, & \alpha_1 &= 0.2,\ \beta_1 = 1,\\
\dot{X}_2 &= \alpha_2 X_1^{-1} - \beta_2 X_2^{0.578151}, & \alpha_2 &= 0.2,\ \beta_2 = 1,\\
\dot{X}_3 &= \alpha_3 X_2^{-1} - \beta_3 X_3^{0.5}, & \alpha_3 &= 0.2,\ \beta_3 = 1.
\end{aligned}$$

From this model we obtain an hybrid automaton with only one discrete location, no transitions and three variables, X_1, X_2, X_3, representing proteins concentrations. For the unique discrete location v we have $Inv(v) \equiv true$, the dynamics in v are defined by the differential equations of the S-System model.

As in the previous case, to obtain a continuous semi-algebraic automaton we approximate the solutions of the differential equations in the discrete location with the corresponding Taylor polynomial of degree two. Consider for instance the differential equation for X_1, we approximate its solution with the following polynomial:

$$X_1' = X_1 + (0.2X_3^{-1} - X_1^{0.5})T + (-0.04X_3^{-2}X_2^{-1} + 0.2X_3^{-1.5} - 0.1X_1^{-0.5}X_3^{-1} + 0.5)T^2/2$$

The automaton we obtain has non-independent dynamics with real exponents, hence we analyse it using the approximated method based on the computation of minimum and maximum values of the linearization of the functions defining the dynamics. We succeed in the simulation of the automaton, but we do not obtain any interesting result.

The analysis of the two models shows that the one obtained from the S-System does not permit to observe the oscillatory behaviour of the repressilator, this because of the approximations introduced for simulation. The other model, instead, results to be less sensitive to approximation and simulating it we can observe the oscillations

in proteins concentrations. This points out that in order to define robust models for biological systems it is important to distinguish from the beginning the discrete from the continuous parts of the systems. Hybrid automata allow to do this and hence to obtain simpler dynamics in each discrete location. Such dynamics are less sensible to the approximations which are necessary to carry out formal analysis.

5 Conclusions

In this paper we presented some experimental results on the reachability problem in semi-algebraic hybrid automata. Our results suggest that even if we try to exploit different techniques and powerful tools, we cannot go *far* enough, without introducing approximations.

However, it is easy to apply some standard, basic, approximation techniques. We showed on the repressilator case study that the approximated results are coherent with the expected behaviour, even when we limit our approximations to the first and second degree, provided that intrinsic discrete nature of the system has been explicitly modeled. In particular, the approximations on the 8-states automaton show the oscillations, while this is not the case if we directly apply our method to the system of differential equations. Intuitively, the system of differential equations implicitly models the discrete nature of the system exploiting more complex dynamics whose simulation requires more sophisticated techniques. The hybrid automaton allows to keep the dynamics more simple and more *robust* to approximations.

References

[1] R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G. J. Pappas, H. Rubin, and J. Schug. Hybrid Modeling and Simulation of Biomolecular Networks. In *Proceedings of Hybrid Systems: Computation and Control (HSCC'01)*, volume 2034 of *LNCS*, pages 19–32. Springer-Verlag, 2001.

[2] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[3] R. Alur, T. Dang, and F. Ivancic. Progress on reachability analysis of hybrid systems using predicate abstraction. In O. Maler and A. Pnueli, editors, *HSCC*, volume 2623 of *LNCS*, pages 4–19. Springer-Verlag, 2003.

[4] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[5] R. Alur, T. A. Henzinger, and Pei-Hsin Ho. Automatic Symbolic Verification of Embedded Systems. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 2–11. IEEE Computer Society Press, 1993.

[6] M. Antoniotti, A. Policriti, N. Ugel, and B. Mishra. Model Building and Model Checking for Biological Processes. *Cell Biochemistry and Biophysics*, 2003.

[7] K.R. Apt and M.G. Wallace. *Constraint Logic Programming using Eclipse*. Cambridge University Press, 2006.

[8] E. Asarin, T. Dang, and O. Maler. The d/dt tool for verification of hybrid systems. In *Proceedings of the Forteenth International Conference on Computer Aided Verification (CAV'02)*, LNCS, pages 365–370, London, UK, 2002. Springer-Verlag.

[9] E. Asarin, T. Dang, O. Maler, and O. Bournez. Approximate Reachability Analysis of Piecewise-Linear Dynamical Systems. In B. Krogh and N. Lynch, editors, *Proceedings of Hybrid Systems: Computation and Control (HSCC'00)*, volume 1790 of *LNCS*, pages 20–31. Springer-Verlag, 2000.

[10] S. Basu. An improved algorithm for quantifier elimination over real closed fields. In *Proceedings of the Thirty-Eighth Annual Symposium on Foundations of Computer Science (FOCS '97)*, pages 56–65, Washington, DC, USA, 1997. IEEE Computer Society Press.

[11] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi. Uppaal - a tool suite for automatic verification of real-time systems. In R. Alur, T. A. Henzinger, and E. D. Sontag, editors, *Hybrid Systems III: Verification and Control, Proceedings of the DIMACS/SYCON Workshop*, volume 1066 of *LNCS*, pages 232–243. Springer-Verlag, 1996.

[12] L. Bortolussi and A. Policriti. Hybrid approximation of Stochastic Concurrent Constraint Programming. In *International Federation of Automatic Control World Congress, (IFAC'08)*, 2008. To appear.

[13] C. W. Brown. QEPCAD B: A program for computing with semi-algebraic sets using CADs. *ACM SIGSAM Bulletin*, 37(4):97–108, 2003.

[14] A. Casagrande, C. Piazza, and B. Mishra. Semi-Algebraic Constant Reset Hybrid Automata -SACoRe. In *Proc. of the 44rd Conference on Decision and Control (CDC'05)*, pages 678–683. IEEE Computer Society Press, 2005.

[15] A. Chutinan and B. Krogh. Verification of Polyhedral-Invariant Hybrid Automata Using Polygonal Flow Pipe Approximations. In F. W. Vaandrager and J. H. van Schuppen, editors, *Proceedings of Hybrid Systems: Computation and Control (HSCC'99)*, volume 1569 of *LNCS*, pages 76–90. Springer-Verlag, 1999.

[16] G. E. Collins. Quantifier Elimination for the Elementary Theory of Real Closed Fields by Cylindrical Algebraic Decomposition. In *Proceedings of the Second GI Conference on Automata Theory and Formal Languages*, volume 33 of *LNCS*, pages 134–183. Springer-Verlag, 1975.

[17] T. Dang and O. Maler. Reachability analysis via face lifting. In T. A. Henzinger and S. Sastry, editors, *HSCC*, volume 1386 of *LNCS*, pages 96–109. Springer-Verlag, 1998.

[18] C. Daws, A. Olivero, S. Tripakis, and S. Yovine. The tool KRONOS. In R. Alur, T. A. Henzinger, and E. D. Sontag, editors, *Hybrid Systems III: Verification and Control, Proceedings of the DIMACS/SYCON Workshop*, volume 1066 of *LNCS*, pages 208–219. Springer-Verlag, 1996.

[19] A. Dolzmann and T. Sturm. REDLOG: Computer algebra meets computer logic. *SIGSAM Bulletin (ACM Special Interest Group on Symbolic and Algebraic Manipulation)*, 31(2):2–9, 1997.

[20] M. Elowitz and S. Leibler. A Synthetic Oscillatory Network of Transcriptional Regulators. *Nature*, 403:335–338, 2000.

[21] R. Ghosh, A. Tiwari, and C. Tomlin. Automated Symbolic Reachability Analysis; with Application to Delta-Notch Signaling Automata. In O. Maler and A. Pnueli, editors, *Proceedings of Hybrid Systems: Computation and Control (HSCC'03)*, volume 2623 of *LNCS*, pages 233–248. Springer-Verlag, 2003.

[22] D. Grigorév and N. Vorobjov. Counting connected components of a semialgebraic set in subexponential time. *Computational Complexity*, 2(2):133–186, 1992.

[23] N. Halbwachs, Y.-E. Proy, and P. Raymond. Verification of linear hybrid systems by means of convex approximations. In B. Le Charlier, editor, *Static Analysis Symposium*, pages 223–237. Springer-Verlag, 1994.

[24] T. A. Henzinger, P. H. Ho, and H. Wong-Toi. HYTECH: A Model Checker for Hybrid Systems. *International Journal on Software Tools for Technology Transfer*, 1(1–2):110–122, 1997.

[25] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? In *Proceedings of the Twenty-Seventh Annual ACM Symposium on the Theory of Computing (STOC '95)*, pages 373–382, New York, NY, USA, 29 May–1 June 1995. ACM Press.

[26] A. B. Kurzhanski and P. Varaiya. On verification of controlled hybrid dynamics through ellipsoidal techniques. In *Proceedings of the 44rd Conference on Decision and Control and European Control Conference (CDC-ECC'05)*, pages 4682–4687, Seville, Spain, December 2005. IEEE Computer Society Press.

[27] G. Lafferriere, G. J. Pappas, and S. Sastry. O-Minimal Hybrid Systems. *Mathematics of Control, Signals, and Systems*, 13:1–21, 2000.

[28] R. Lanotte and S. Tini. Taylor approximation for hybrid systems. *Inf. Comput.*, 205(11):1575–1607, 2007.

[29] V. Mysore and B. Mishra. Algorithmic Algebraic Model Checking III: Approximate Methods. *Electronic Notes in Theoretical Computer Science*, 149(1):61–77, 2006.

[30] V. Mysore, C. Piazza, and B. Mishra. Algorithmic Algebraic Model Checking I: Decidability of Semi-Algebraic Model Checking and its Applications to Systems Biology. In Y. Tsay D. A. Peled, editor, *Proceedings Third International Symposium on Automated Technology for Verification and Analysis (ATVA 2005)*, number 3707 in LNCS, pages 217–233. Springer-Verlag, October 2005.

[31] C. Piazza, M. Antoniotti, V. Mysore, A. Policriti, F. Winkler, and B. Mishra. Algorithmic Algebraic Model Checking I: Challenges from Systems Biology. In K. Etessami and S. K. Rajamani, editors, *Proceedings Computer Aided Verification (CAV'05)*, number 3576 in LNCS, pages 5–19. Springer-Verlag, 2005.

[32] A. Puri and P. Varaiya. Decidability of hybrid systems with rectangular differential inclusions. In D. L. Dill, editor, *Proceedings of International Conference on Computer Aided Verification (CAV'94)*, volume 818 of *LNCS*, pages 95–104. Springer-Verlag, 1994.

[33] S. Ratschan. Efficient Solving of Quantified Inequality Constraints over the Real Numbers. *ACM Transactions on Computational Logic*, 7(4):723–748, 2006.

[34] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. Part III: Quantifier elimination. *Journal of Symbolic Computation*, 13(3):329–352, 1992.

[35] B. I. Silva and B. H. Krogh. Formal verification of hybrid system using checkmate: A case study. In *Proceedings of the American Control Conference 2000 (ACC'00)*, pages 678–683, Chicago, Illinois, June 2000. IEEE Computer Society Press.

[36] T. Sturm. Quantifier Elimination-Based Constraint Logic Programming. Technical Report MIP-0202, Fakultät für Mathematik und Informatik, Universität Passau, 2002.

[37] A. Tarski. *A Decision Method for Elementary Algebra and Geometry.* Univ. California Press, 1951.

[38] E. O. Voit. *Computational Analysis of Biochemical Systems. A Pratical Guide for Biochemists and Molecular Biologists.* Cambridge University Press, 2000.