Noname manuscript No.
(will be inserted by the editor)

Quantifier-free encoding of invariants for hybrid systems*

A. Cimatti · S. Mover · S. Tonetta

the date of receipt and acceptance should be inserted later

Abstract Hybrid systems are a clean modeling framework for embedded systems, which feature integrated discrete and continuous dynamics. A well-known source of complexity comes from the time invariants, which represent an implicit quantification of a constraint over all time points of a continuous transition.

Emerging techniques based on Satisfiability Modulo Theory (SMT) have been found promising for the verification and validation of hybrid systems because they combine discrete reasoning with solvers for first-order theories. However, these techniques are efficient for quantifier-free theories and the current approaches have so far either ignored time invariants or have been limited to hybrid systems with linear constraints.

In this paper, we propose a new method that encodes a class of hybrid systems into transition systems with quantifier-free formulas. The method does not rely on expensive quantifier elimination procedures. Rather, it exploits the sequential nature of the transition system to split the continuous evolution enforcing the invariants on the discrete time points. This way, we can encode all hybrid systems whose invariants can be expressed in terms of polynomial constraints. This pushes the application of SMT-based techniques beyond the standard linear case.

1 Introduction

Embedded systems (e.g. control systems for railways, avionics, and space) feature the interaction of discrete systems with the environment by means of controlled and monitored variables that evolve continuously in time. The validation and verification of embedded systems designs must often take into account a model of the continuous evolution of such variables. *Hybrid systems* [3] are a clean modeling framework for embedded systems because they exhibit both continuous transitions ruled by flow conditions and discrete changes represented with logical formulas.

Alessandro Cimatti, Sergio Mover, Stefano Tonetta (Fondazione Bruno Kessler, Trento, Italy)
E-mail: {cimatti,mover,tonettas}@fbk.eu

*This paper presents in a coherent and expanded form material that appears in the conference venue [16].

A fundamental step in the design of these systems is the validation and verification of the models, performed by checking properties such as invariants or reachability. In spite of the undecidability of these problems, several verification techniques have been developed and have proved to be applicable in a wide number of cases. Among these techniques, common approaches are the computation of the reachable states, and the use of abstraction or deduction systems (see [2] for a recent survey).

An emerging approach to the verification of hybrid systems is the application of verification techniques based on SMT [6]. The hybrid system is encoded into a symbolic transition system and reachability problems are represented by means of first-order formulas. The encoding allows the application of general-purpose SAT-based verification techniques such as Bounded Model Checking (BMC) [7], interpolation-based model checking [31], k-induction [38], and predicate abstraction [24], and combination thereof [40]. Examples of such SMT-based approaches are [5,1,28,26,20,29,21,27]. Specific techniques have also been proposed for networks of hybrid systems [10,13,15], and for requirements [17]. Also thanks to the strong progress in the field of SMT, these approaches are increasingly applied in real settings (e.g. the design of complex space systems [8,9,41]).

A well-known problem of this approach is the encoding of invariants. In order to preserve the semantics of the hybrid system, the formula representing a continuous (timed) transition between two time points t and t' must guarantee that the invariant holds along all points of the implicit continuous evolution between the state $s(t)$ and the state $s(t')$. A straightforward approach would create a quantified formula which treats the invariant as a formula $Inv(t)$ over the variable representing real time and quantifies the formula along all time points of the timed transition, i.e., $\forall \epsilon \in [t, t'], Inv(\epsilon)$. In general, it is an open question how to handle such quantifiers (see for example [1,21]): the elimination of quantifiers is in general not possible, and when the elimination is theoretically possible (such as in the case of the theory of reals, i.e., polynomial constraints) it is in practice not feasible beyond the quadratic case. Only in particular cases (such as when the continuous evolution of variables is expressed as a linear function of time, and Inv is convex), the encoding is equivalent to the quantifier-free formula $Inv(t) \wedge Inv(t')$.

In this paper, we propose a new approach to efficiently encode invariants as quantifier-free formulas. Intuitively, the encoding can be thought of as generalizing the linear case, forcing the invariant before and after the timed transition ($Inv(t) \wedge Inv(t')$), and imposing the derivative of the invariant to be constant in sign throughout the timed transition. This reduces the invariant to a quantified formula over the derivatives of the continuous variables. Applying these reductions recursively, in some cases, one may obtain a quantifier-free encoding. In particular, we prove that this holds for hybrid systems with polynomial dynamics, where the continuous evolution is explicitly expressed by polynomial functions of time, since the derivatives eventually reduce to zero. Then, we provide a quantifier-free encoding for two different classes of hybrid systems with linear ODEs, via a reduction to the polynomial dynamics case. Finally, we show that it is possible to obtain a quantifier-free encoding also in interesting non-linear cases of hybrid automata with transcendental dynamics. Overall, a key contribution of the paper is to enable the application of SMT-based verification techniques to a broader class of hybrid systems.

The approach has been implemented and evaluated on a set of benchmarks. The results show that the SMT solvers for non-linear arithmetic are still not able to handle large problems. However, the proposed technique allows us to solve problems where an abstraction that simply ignores the invariants is too coarse to guarantee soundness and completeness. Moreover, the results of the paper should stimulate new research in the field.

The rest of this paper is structured as follows. In Sec. 2 we present some background on symbolic transition and hybrid systems. In Sec. 3, we show how to reduce the encoding problem to the case where the invariants contain only atomic formulas, removing disjunctions from the universally quantified formulas. In Sec. 4, we show how to encode the (implicitly quantified) invariants into quantifier-free formulas. In Sec. 5, we present the encoding of hybrid systems with polynomial dynamics. In Sec. 6, we show how to handle two classes of hybrid systems with linear ODE. A comparison with related work is described in Sec. 7, whilst the experimental evaluation is presented in Sec. 8. In Sec. 9 we draw some conclusions, and outline directions for future work.

2 Background

2.1 First-order Transition Systems

Given a set V of variables, we denote with $V', \dot{V}, V^0, V^1, \ldots$ copies of such set. Given a first-order signature Σ, a first-order Σ-Transition System (TS) is a tuple $S = \langle V, Init, Inv, Trans \rangle$ such that:

- V is a set of variables;
- $Init$ is a first-order Σ-formula over V (called initial condition);
- Inv is a first-order Σ-formula over V (called invariant condition);
- $Trans$ is a first-order Σ-formula over $V \cup V'$ (called transition condition).

Let $\Sigma_{\mathbb{R}}$ be the standard signature of the real ordered field. In the following, we will consider signatures Σ that are extensions of $\Sigma_{\mathbb{R}}$, the structure $\overline{\mathbb{R}}$ of the real ordered field extended with transcendental functions such as the exponential and the trigonometric functions, and formulas will be interpreted in an appropriate extension of the first-order theory of the real numbers for such structure $\overline{\mathbb{R}}$.

A *state* s is an assignment to the variables V. We denote with $s', \dot{s}, s^0, s^1, \ldots$ the corresponding assignment to the copy $V', \dot{V}, V^0, V^1, \ldots$ of V.

A sequence $s_0, s_1, \ldots, s_k$ of states is a model (also called *path*) of the transition system $S = \langle V, Init, Inv, Trans \rangle$ iff:

- s_0 satisfies $Init$;
- for every $0 \leq i \leq k$, s_i satisfies Inv;
- for every $0 \leq i < k$, s_i, s'_{i+1} satisfy $Trans$.

2.2 Hybrid traces

We denote with $\dot{f}$ the first derivative of a real function f. Let I be an interval of $\mathbb{R}$ or $\mathbb{N}$; we denote with $le(I)$ and $ue(I)$ the lower and upper endpoints of I, respectively.

Hybrid traces [18,17] describe the evolution of variables in every point of time. Such evolution is allowed to have a countable number of discontinuous points corresponding to changes in the discrete part of the model.

A *hybrid trace* over discrete variables V and continuous variables X is a sequence $\langle \overline{f}, \overline{I} \rangle := \langle f_0, I_0 \rangle, \langle f_1, I_1 \rangle, \ldots, \langle f_k, I_k \rangle$ such that, for all i, $0 \leq i \leq k$,

- the intervals are adjacent, i.e. $ue(I_i) = le(I_{i+1})$;
- $le(I_0) = 0$ and I_k is right closed;
- $f_i : V \cup X \to \mathbb{R} \to \mathbb{R}$ is a function such that, for all $x \in X$, $f_i(x)$ is differentiable, and for all $v \in V$, $f_i(v)$ is constant;
- if I_i is left open [right open] and $le(I_i) = t$ [$ue(I_i) = t$] then, for all $v \in V \cup X$, $f_i(v)(t) = f_{i-1}(v)(t)$, [$f_i(v)(t) = f_{i+1}(v)(t)$].

We say that a trace is a sampling refinement of another one if it has been obtained by splitting an open interval into two parts by adding a sampling point in the middle [18]. A *partitioning function* μ is a sequence $\mu_0, \mu_1, \mu_2, \ldots$ of non-empty, adjacent and disjoint intervals of $\mathbb{N}$ partitioning $\mathbb{N}$. Formally, $\bigcup_{i \in \mathbb{N}} \mu_i = \mathbb{N}$ and $ue(\mu_i) = le(\mu_{i+1}) - 1$. A hybrid trace $\langle \overline{f}', \overline{I}' \rangle$ is a *sampling refinement* of $\langle \overline{f}, \overline{I} \rangle$ (denoted with $\langle \overline{f}', \overline{I}' \rangle \preceq \langle \overline{f}, \overline{I} \rangle$) iff, there exists a partitioning μ such that for all $i \in \mathbb{N}$, $I_i = \bigcup_{j \in \mu_i} I'_j$ and, for all $j \in \mu_j$, $f'_j = f_i$. We extend the relation to set L_1 and L_2 of traces as follows: $L_1 \preceq L_2$ iff for every trace $\sigma_2 \in L_2$ there exists $\sigma_1 \in L_1$ such that $\sigma_1 \preceq \sigma_2$.

We assume that the evolution of predicates along time have the finite variability property: we say that a predicate $P(t)$ over a real variable t has *finite variability* [36] iff for any bounded interval J there exists a finite sequence of real numbers $t_0 < \ldots < t_n$ such that $t_0 = le(J)$, $t_n = ue(J)$, and for all $i \in [1, n]$, either for all $\epsilon \in (t_{i-1}, t_i)$, $P(\epsilon)$ or for all $\epsilon \in (t_{i-1}, t_i)$, $\neg P(\epsilon)$. The last condition means that the predicate is constant in the interval (t_{i-1}, t_i). If P is in the form $g(t) \bowtie 0$ with g continuous and $\bowtie \in \{\geq, \leq, <, >\}$, in the points in which P changes value, $g(t) = 0$. Thus, $g \bowtie 0$ has finite variability iff for any bounded interval J there exists a finite sequence of real numbers $t_0 < \ldots < t_n$ such that $t_0 = le(J)$, $t_n = ue(J)$, and for all $i \in [1, n]$, either for all $\epsilon \in [t_{i-1}, t_i]$, $g(\epsilon) \geq 0$ or for all $\epsilon \in [t_{i-1}, t_i]$, $g(\epsilon) \leq 0$. We denote this condition with $\mathit{Constant}(P, t_{i-1}, t_i)$.

Proposition 1 *Assuming that a predicate P has finite variability, for every hybrid trace σ, there exists a sampling refinement of σ for which which P is constant in the open part of every interval.*

2.3 Hybrid systems

Hybrid systems [3] extend transition systems with continuous dynamics. A *Hybrid System* (HS) is a tuple $\langle V, X, Init, Trans, Inv, Flow \rangle$ where:

- V is the set of discrete variables,
- X is the set of continuous variables,
- $Init$ is a $\Sigma_{\mathbb{R}}$-formula over $V \cup X$ (called the initial condition);
- Inv is a $\Sigma_{\mathbb{R}}$-formula over $V \cup X$ (called the invariant condition).
- $Trans$ is a $\Sigma_{\mathbb{R}}$-formula over $V \cup X \cup V' \cup X'$ (called the transition condition);
- $Flow$ is a $\Sigma_{\mathbb{R}}$-formula over $V \cup X \cup \dot{X}$ (called the flow condition).

Given a hybrid trace $\langle f_0, I_0\rangle, \langle f_1, I_1\rangle, \ldots, \langle f_k, I_k\rangle$, we denote with $s_{f_i}(t)$ the state assigning to every variable $v \in V \cup X$ the value $f_i(v)(t)$ and with $\dot{s}_{f_i}(t)$ the assignment that maps every variable $v \in X$ with the value $\dot{f}_i(v)(t)$.

A hybrid trace $\langle f_0, I_0\rangle, \langle f_1, I_1\rangle, \ldots, \langle f_k, I_k\rangle$ is a model (also called *path*) of the HS $S = \langle V, X, Init, Inv, Trans, Flow\rangle$ iff:

- $s_{f_0}(0)$ satisfies $Init$;
- for every $0 \leq i \leq k$, for all $t \in I_i$, $s_{f_i}(t)$ satisfies Inv;
- for every $0 \leq i < k$, if I_i is right closed with $ue(I_i) = t$ and I_{i+1} is left closed with $le(I_{i+1}) = t'$, then $s_{f_i}(t), s'_{f_{i+1}}(t')$ satisfies $Trans$;
- for every $0 \leq i \leq k$, for all $t \in I_i$, $s_{f_i}(t), \dot{s}_{f_i}(t)$ satisfy $Flow$.

The *language* $L(S)$ is the set of models of S.

Proposition 2 *A sampling refinement of a path of an HS S is also a path of S.*

Intuitively, sampling refinement just splits an interval into sub-intervals and therefore does not change either the initial state or the discrete transitions. Thus, the conditions remain satisfied by the corresponding points.

Sampling refinement preserves reachability properties in the sense that if $L' \preceq L(S)$ then there exists a trace in L' reaching a condition ϕ iff there exists a trace in $L(S)$ reaching ϕ (similarly for LTL properties without next operators [18] and HRELTL properties [17]).

Remark 1 In the above definition, the flow conditions are general predicates over the derivatives of X. In the following, we are considering HSs with continuous dynamics described by ODEs in form $\dot{X} = F(X)$ (i.e., for all $x \in X$, $\dot{x} = F_x(X)$). Then, we assume that the system of ODEs admits a primitive solution $f(V, t)$, which is uniquely determined by the state at the beginning of the timed transition.

Remark 2 We clarify the nomenclature used in this paper when referring to the different kinds of hybrid systems that we considered.

- Hybrid systems with linear (or non-linear) constraints (see, e.g., [3, 25]): linear and non-linear hybrid automata, where the flow condition is given by symbolic constraints over the derivatives of continuous variables.
- Hybrid systems with linear (or non-linear) ODE (see, e.g., [4, **?**,39, 23]): following the literature on control theory, linear and non-linear hybrid systems where the flow condition is defined by a system of linear or non-linear Ordinary Differential Equations (ODE).
- Hybrid systems with polynomial (or non-linear) dynamics (see, e.g., [22, 12, 34]): hybrid systems such that the continuous evolution is described with a function of time, thus without using derivatives.

Our definition of hybrid systems is general enough to cover the first two cases. However, our results also applies to the third case, as such dynamics provide the explicit solution for a system of ODEs, as we require.

2.4 Encoding of hybrid into transition systems

In this section, we show a standard encoding of HSs into a transition system with formulas over the reals. In general, for encoding, we mean a transition system

that preserves the properties of interest. In this paper, we say that the transition system is an encoding of a HS if it represents its language or a sampling refinement thereof (thus preserving for example reachability).

In this encoding, we assume that the system of ODEs admits a primitive solution $f(V,t)$, which is uniquely determined by the state at the beginning of the timed transition. Moreover, we assume that the time intervals of the hybrid traces satisfying the HSs are all in the form either $[t_1,t_2)$ (i.e., left closed, right open) or $[t_1,t_1]$ (i.e., singular intervals). This simplifies the encoding but a more general encoding is possible (see for example [17]). Note also that the restriction does not affect the validity of Proposition 1, which regards only the open parts of the intervals.

Theorem 1 *Given a HS S, there exists a TS S_D such that there exists a one-to-one mapping between the paths of S and the paths of S_D.*

From now on, we will call S_D the *encoding* of S.

Proof (Sketched proof) We encode a HS S in the TS $S_D = \langle V_D, Init_D, Inv_D, Trans_D\rangle$ where:

- $V_D := V \cup X \cup \{t\}$
  (t is a real variable that stores the current real time of the system).
- $Init_D := t = 0 \wedge Init$.
- $Inv_D := Inv$
  (note that this does not guarantee that the invariants of S hold for the entire duration of a continuous transition. This is taken into account in $Trans_D$).
- $Trans_D :=$ Timed $\vee$ Untimed where
  - Timed $:= t' > t \wedge V' = V \wedge X' = f(V \cup X, t') \wedge \forall \epsilon \in [t,t'], Inv(V, f(\epsilon))$
  - Untimed $:= t' = t \wedge Trans(V, X, V', X')$.

Let the hybrid trace $\langle f_0, I_0\rangle, \langle f_1, I_1\rangle, \ldots, \langle f_k, I_k\rangle$ be a path of S. Then, the sequence of states $f_0(le(I_0)), f_1(le(I_1)), \ldots, f_k(le(I_k))$ is a path of S_D.

Let the sequence $s_0, s_1, \ldots, s_k$ be a path of S_D. Let us consider, for all $i \in [1,k]$, $f_i(v)(t) = f(s_i,t)(v)$. Let us define $I_i := [s_i(t), s_{i+1}(t))$ if $i < k$ and $s_{i+1}(t) > s_i(t)$, $I_i := [s_i(t), s_i(t)]$ if $i < k$ and $s_{i+1}(t) = s_i(t)$ or if $i = k$. Then, the hybrid trace $\langle f_0, I_0\rangle, \langle f_1, I_1\rangle, \ldots, \langle f_k, I_k\rangle$ is a path of S.

Remark 3 Notice that the timed transition involves a quantified sub-formula to encode that the invariant holds along each instant of the continuous evolution. This is an issue for using standard SMT solvers which typically handle quantifier-free formulas or are not complete for quantifiers (even if the full theory with quantifiers is theoretically decidable). When the primitive solution is known and is expressed in the theory of reals (a polynomial), the quantifier can be removed to yield an equivalent quantifier-free encoding. However, in practice, this solution is not feasible beyond the quadratic case.

Remark 4 In the formula $\forall \epsilon \in [t,t'], Inv(V, f(\epsilon))$ representing the invariant during the timed transition, the quantifier ranges over the closed interval $[t,t']$. However, since Inv is also forced by the invariant of S_D, we can safely replace the interval with its open version (t,t') obtaining an equivalent transition system (with the very same paths). The timed transition in this variant would be:
Timed$' := t' > t \wedge V' = V \wedge X' = f(V \cup X, t') \wedge \forall \epsilon \in (t,t'), Inv(V, f(\epsilon))$

This will be useful when dealing with disjunctive invariants. In the following, we will clarify which encoding we are considering.

Remark 5 It is usually very useful to strictly alternate timed and discrete transitions to simplify the encoding and improve the search (see e.g. [1]). The encoding of Theorem 1 does not force such alternation, to enable other forms of simplification. In the following, we will clarify when we use alternation.

3 Removing quantified disjunctions from the invariants

In this section we describe how to remove disjunctions from the invariants, obtaining an equivalent encoding where the quantified formulas contain only atomic predicates. The transformation relies on the encoding of hybrid systems with quantifiers over open intervals (see Remark 4). Then, in the next Section we will show how to remove the quantifiers in both the open and the closed intervals case. Note that the existing encodings of hybrid automata into infinite-state transition systems ignore the issue and assume the convexity of the invariant condition.

We reduce the quantification over a disjunctive invariant into a disjunction of quantifications. While this is not correct in general, it is possible due to the particular position of the quantified sub-formula in the transition condition. After the reduction we guarantee that the quantified formula in $Trans$ is atomic, allowing us to remove the quantifiers.

Suppose we have a disjuntive invariant $\phi(\epsilon) \vee \psi(\epsilon)$. In our case we can distribute the universal quantifier in $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \psi(\epsilon)$ over the disjunction, obtaining the formula $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \forall \epsilon \in (t, t'), \psi(\epsilon)$. The following theorem proves the correctness of the transformation.

Theorem 2 *Assuming that the predicates ϕ and ψ have finite variability, if we replace a formula $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \psi(\epsilon)$ with $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \forall \epsilon \in (t, t'), \psi(\epsilon)$ inside $Trans_D$, we obtain the encoding of a sampling refinement to the original HS.*

Proof Clearly, $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \forall \epsilon \in (t, t'), \psi(\epsilon)$ implies $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \psi(\epsilon)$. The opposite does not hold in general. However, consider a hybrid trace $\langle \overline{f}, \overline{I} \rangle := \langle f_0, I_0 \rangle, \langle f_1, I_1 \rangle, \ldots, \langle f_k, I_k \rangle$ which is a path of a HS S. Assuming that the predicates ϕ and ψ have finite variability, we can refine the hybrid trace into a new hybrid trace $\langle \overline{f'}, \overline{I'} \rangle' := \langle f'_0, I_{;0} \rangle, \langle f'_1, I'_1 \rangle, \ldots, \langle f'_l, I'_l \rangle$ in which ϕ and ψ are constant in every interval. The new hybrid trace also satisfies S by Proposition 2 and thus the corresponding discrete trace $s_0, \ldots, s_l$ satisfies its encoding S'_D. At every i, if s_i satisfies $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \psi(\epsilon)$, then $f(s_i, \epsilon)$ satisfies $\phi \vee \psi$ for all $\epsilon \in (t, t') = (le(I'_i), ue(I'_i))$, and thus either ϕ or ψ (since ϕ and ψ are constant in the open part of I'_i). Therefore the discrete trace satisfies also the encoding with $\forall \epsilon \in (t, t'), \phi(\epsilon) \vee \forall \epsilon \in (t, t'), \psi(\epsilon)$.

The effect of the encoding on the paths of the transition system is shown in Figure 1. In practice, the encoding of the transition system without disjunctions splits the continuous transition every time the valuation of ϕ or ψ changes, instead of allowing a single continuous transition where $\phi \vee \psi$ holds.

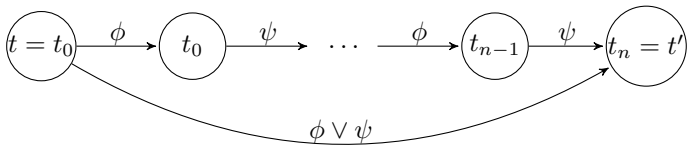

Fig. 1 Effect on a path of the encoding without disjunctions.

4 Removing quantifiers from the invariants

4.1 Reduction to flow invariants

In this section we present the main theorem of the paper. The goal of the theorem is to reduce the quantified formula of an invariant to a quantified formula over its derivatives. In some cases, this simplifies the quantified formula.

The following theorems assume the finite variability of predicates of the derivatives. Many functions have this property, in particular polynomials and some simple transcendental functions.

Theorem 3 *If $g : \mathbb{R} \to \mathbb{R}$ is a differentiable function and $\dot{g} \bowtie 0$ ($\bowtie \in \{\geq, >, \leq, <\}$) has finite variability, then $\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0$ iff there exists a finite sequence of real numbers $t = t_0 < \ldots < t_n = t'$ such that $\bigwedge_{0 \leq i \leq n} g(t_i) \bowtie 0 \wedge \bigwedge_{0 < i \leq n}$ Constant$(\dot{g} \geq 0, t_{i-1}, t_i)$.*

Proof Let us assume that $\bowtie \in \{\geq, >\}$.

($\Rightarrow$) Since $\dot{g} \bowtie 0$ has finite variability, there exists a finite sequence of real numbers $t_0 = t < \ldots < t_n = t'$ such that $\bigwedge_{0 < i \leq n}$ Constant$(\dot{g} \geq 0, t_{i-1}, t_i)$ by definition. Moreover, since $\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0$, $g \bowtie 0$ holds also in the time points $t_0, \ldots, t_n$.

($\Leftarrow$) Assume by contradiction that there exists $t_b \in [t, t']$ such that $g(t_b) \bowtie 0$ is false. Since $\bigwedge_{0 \leq i \leq n} g(t_i) \bowtie 0$, there exists $i \in [1, n]$ such that $t_b \in (t_{i-1}, t_i)$. Since g is differentiable, by the mean value theorem, there exists a point $t'_b \in (t_{i-1}, t_b)$ such that $\dot{g}(t'_b) = \frac{g(t_b) - g(t_{i-1})}{(t_b - t_{i-1})}$ and therefore $\dot{g}(t'_b) < 0$. Similarly, there exists a point $t''_b \in (t_b, t_i)$ such that $\dot{g}(t''_b) = \frac{g(t_i) - g(t_b)}{(t_i - t_b)}$ and therefore $\dot{g}(t''_b) > 0$. Thus, $\dot{g}$ is not constant over (t_{i-1}, t_i) contradicting the hypothesis. We conclude that $\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0$.

The cases in which $\bowtie \in \{\leq, <\}$ can be proved similarly.

The intuition behind the theorem is simple. While we can easily encode that the invariant holds at some precise point (e.g. t, t') it is harder to impose the same condition along an interval without the use of quantifiers. However, we exploit the sign of $\dot{g}$ to infer the behavior of g in $[t, t']$ (i.e. if $\dot{g} > 0$, resp. $\dot{g} < 0$, resp. $\dot{g} = 0$, then g increases, resp. decreases, resp. is constant). Since g is finite variable, we can divide the interval $[t, t']$ in a finite sequence of intervals where the sign of the derivative is constant. If the invariant does not hold in all the endpoints of the intervals, then there exists a point in $[t, t']$ where the invariant does not hold. Otherwise, by the fact that the sign of the derivative is constant we know that g just increases, decreases or is constant in the interval. Thus, the valuation of $g \bowtie 0$ does not change in the interval.

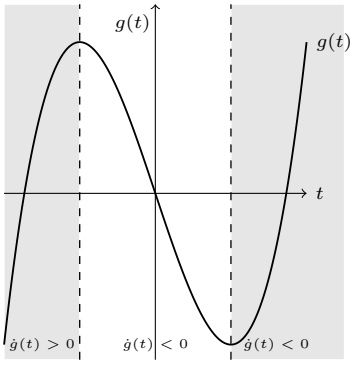

Fig. 2 Plot of the function $t^3 - 3t$. The plot is grey if $\dot{g}(t) > 0$ and white when $\dot{g}(t) < 0$.

Example 1 Consider the function $g(t) = t^3 - 3t$ in the interval $[-2, 2]$ (it is plotted in Figure 2) and the invariant $g(t) \geq 0$. The derivative $\dot{g}(t) < 0$ in $[-2, -1)$ and $(1, 2]$, $\dot{g}(t) > 0$ in $(-1, -1)$, $\dot{g}(t) = 0$ in $[-1, -1]$ and $[1, 1]$. Now consider the formula $\forall \epsilon \in [-2, 0], g(\epsilon) \geq 0$. We remove the quantifier considering the intervals where the derivative is constant: $g(-2) \geq 0 \wedge g(-1) \geq 0 \wedge g(0) \geq 0 \wedge$ $Constant(\dot{g} \geq 0, -2, -1)\, Constant(\dot{g} \geq 0, -1, 0)$. Note that in practice we do not fix the intervals where the derivative is constant, but they will be found automatically by the SMT solver.

When the predicate is an equality, the reduction is simpler.

Corollary 1 *If $g : \mathbb{R} \to \mathbb{R}$ is a differentiable function and $\dot{g} = 0$ has finite variability, then $\forall \epsilon \in [t, t'], g(\epsilon) = 0$ iff $g(t) = 0 \wedge g(t') = 0 \wedge \forall \epsilon \in [t, t'], \dot{g}(\epsilon) = 0$.*

The quantifier can be removed even if the interval of quantification is open.

Theorem 4 *If $g : \mathbb{R} \to \mathbb{R}$ is a differentiable function and $\dot{g} \bowtie 0$ ($\bowtie \in \{\geq, >\}$) has finite variability, then $\forall \epsilon \in (t, t'), g \bowtie 0$ iff there exists a finite set of real numbers $t = t_0 < \ldots < t_n = t'$ such that $g(t) \geq 0 \wedge g(t') \geq 0 \wedge \bigwedge_{0<i<n} g(t_i) \bowtie 0 \wedge \bigwedge_{0<i\leq n} Constant(\dot{g} \geq 0, t_{i-1}, t_i)$ if $\bowtie = \geq$, $g(t) \geq 0 \wedge g(t') \geq 0 \wedge \bigwedge_{0<i<n} g(t_i) \bowtie 0 \wedge \bigwedge_{0<i\leq n} Constant(\dot{g} \geq 0, t_{i-1}, t_i) \wedge (g(t) = 0 \to \dot{g}(t) > 0) \wedge (g(t') = 0 \to \dot{g}(t) < 0)$, if $\bowtie = >$.*

Proof ($\Rightarrow$) Since $\dot{g} \bowtie 0$ has finite variability, there exists a finite set of real numbers $t = t_0 < \ldots < t_n = t'$ such that $\bigwedge_{0<i\leq n} Constant(\dot{g} \geq 0, t_{i-1}, t_i)$ by definition. Moreover, since $\forall \epsilon \in (t, t'), g \bowtie 0$, $g \bowtie 0$ holds also in the time points $t_1, \ldots, t_{n-1}$. $g(t) \geq 0$ and $g(t') \geq 0$ for the continuity of g. Finally, $(g(t) = 0 \to \dot{g}(t) > 0) \wedge (g(t') = 0 \to \dot{g}(t) < 0)$, if $\bowtie = >$.

($\Leftarrow$) Assume by contradiction that there exists $t_b \in (t, t')$ such that $g(t_b) \bowtie 0$ is false. Since $\bigwedge_{0<i<n} g(t_i) \bowtie 0$, there exists $i \in [1, n]$ such that $t_b \in (t_{i-1}, t_i)$.

Let us consider first the cases in which $\bowtie = \geq$ or $i \in [2, n-1]$ or $i = 1$ and $g(t) \bowtie 0$ or $i = n$ and $g(t') \bowtie 0$. Since g is differentiable, for the mean value theorem, there exists a point $t'_b \in (t_{i-1}, t_b)$ such that $\dot{g}(t'_b) = \frac{g(t_b) - g(t_{i-1})}{(t_b - t_{i-1})}$ and therefore $\dot{g}(t'_b) < 0$. Similarly, there exists a point $t''_b \in (t_b, t_i)$ such that $\dot{g}(t''_b) = \frac{g(t_i) - g(t_b)}{(t_i - t_b)}$ and therefore $\dot{g}(t''_b) > 0$. Thus, $\dot{g}$ is not constant over (t_{i-1}, t_i) contradicting the hypothesis.

Let us now consider the case in which $\bowtie = >$, $i = 1$ and $g(t) = 0$ (the case $i = n$ and $g(t') = 0$ is similar). By hypothesis, $\dot{g}(t) > 0$. Thus, there exists $t_0 \in (t_{i-1}, t_b)$ such that $g(t_0) > 0$. As before there exists a point $t'_b \in (t_0, t_b)$ such that $\dot{g}(t'_b) = \frac{g(t_b)-g(t_0)}{(t_b-t_0)}$ and therefore $\dot{g}(t'_b) < 0$. Similarly, there exists a point $t''_b \in (t_b, t_i)$ such that $\dot{g}(t''_b) = \frac{g(t_i)-g(t_b)}{(t_i-t_b)}$ and therefore $\dot{g}(t''_b) > 0$. Therefore $\dot{g}$ is not constant over (t_{i-1}, t_i) contradicting the hypothesis.

We conclude that $\forall \epsilon \in (t, t'), g(\epsilon) \bowtie 0$.

Hereafter, unless otherwise specified, we assume that every universal quantifier occurs positively in $Trans_D$ and that it is in the form $\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0$, with $\bowtie \in \{<, \leq, >, \geq, =\}$. As shown by the previous theorem, the assumption does not limit the approach, but it simplifies the presentation of the results of the paper.

The definition of *Constant*() contains quantified sub-formulas in the form $\forall \epsilon \in [t, t'], \dot{g} \bowtie 0$. Therefore, the reduction can be iterated trying to remove the quantifiers.

Theorem 3 can be used to simplify the encoding of the invariant of an HS. Let the invariant be in the form $g(X) \bowtie 0$ ($\bowtie \in \{\geq, \leq, >, <, =\}$). Let $f : \mathbb{R} \to \mathbb{R}^{|X|}$ be the solution of the flow condition. If f and g are differentiable functions and $\frac{d}{dt}(g \circ f) \bowtie 0$ has finite variability, then $\forall \epsilon \in [t, t'], g(f(\epsilon)) \bowtie 0$ iff there exists a finite sequence of real numbers $t_0 = t < \ldots < t_n = t'$ such that $\bigwedge_{0 \leq i \leq n} g(f(t_i)) \bowtie 0 \wedge \bigwedge_{0 < i \leq n} \mathit{Constant}(\frac{d}{dt}(g \circ f) \geq 0, t_{i-1}, t_i)$.

The geometrical interpretation of $\frac{d}{dt}(g \circ f)$ is the scalar product of the gradient of the curve g and the derivative vector $\dot{f}$: in fact, $\frac{d}{dt} g(f(t)) = \bigtriangledown g \cdot \dot{f}$ where $\bigtriangledown g = \langle \frac{\partial g}{\partial x_1}, \ldots, \frac{\partial g}{\partial x_n} \rangle$. Therefore, in the theorem, the condition of $\dot{g} \geq 0$ of being constant in the interval means that the function f is uniformly getting closer to (or farther from) the curve g in that interval.

As a side note, in the case of ODEs $\dot{X} = F(X)$, the new quantified formula $\forall \epsilon \in [t, t']$, $\frac{d}{dt}(g \circ f) \geq 0$ is equivalent to the invariant $\bigtriangledown g \cdot F \geq 0$. Thus, the reduction can be also applied without need of the primitive solutions.

In the case that the invariants are polynomial and the continuous variables are polynomial functions of time, the derivative will eventually reduce to zero.

4.2 Applications

4.2.1 Application to polynomial hybrid automata

We consider the class of HS where the invariants and the primitive solution of the ODEs are polynomial functions of time (see also [22]). The polynomial may contain some discrete variables as coefficients to account for uncertainties in the inputs, model parameters, etc. Note that several classes of HS with linear ODE can be expressed as a polynomial hybrid automaton, since the solution to the linear system of ODEs can be expressed as a quantifier free formula in the theory of reals [30]. We describe a quantifier-free encoding for some classes of linear hybrid systems in Section 6.

Theorem 5 *The invariant of a polynomial hybrid automaton can be encoded with a quantifier-free formula.*

Proof In the case of polynomial hybrid automata, the invariant $g \bowtie 0$ is encoded into a formula in the form $\forall \epsilon \in [t, t'], g(f(\epsilon)) \bowtie 0$. If g and f are polynomials, $g \circ f$ is also a polynomial. The derivative of a polynomial has a lower degree than the polynomial itself. Thus, at every application of Theorem 3, the degree of the polynomial inside the quantifier strictly decreases. Thus, after a finite number of applications of the theorem, we obtain a quantifier-free formula.

Example 2 Let us consider the classical example of the bouncing ball. Suppose the ball moves in two dimensions x and y, where x is the horizontal coordinate, with $\dot{x} = v_0$, and y is the vertical coordinate, with $\dot{y} = w$ and $\dot{w} = -g$. Thus, the primitive solution is $x(t) = v_0 t + x_0$, $y(t) = -\frac{g}{2}t^2 + w_0 t + y_0$, and $w(t) = -gt + w_0$. Suppose the ball is bouncing on a parabolic hill, a curved surface with equation $y + ax^2 + bx + c = 0$. The invariant of the continuous transition is $y + ax^2 + bx + c \geq 0$ and its encoding is $\forall \epsilon \in [t, t'], y(\epsilon) + ax^2(\epsilon) + bx(\epsilon) + c \geq 0$, which is quadratic in ϵ. After applying the Theorem 3 twice, we obtain the following quantifier-free formula: $y(t) + ax^2(t) + bx(t) + c \geq 0 \wedge$
$y(t_1) + ax^2(t_1) + bx(t_1) + c \geq 0 \wedge$
$y(t') + ax^2(t') + bx(t') + c \geq 0 \wedge$
$((w(t) + 2av_0x(t) + bv_0 \geq 0 \wedge w(t_1) + 2av_0x(t_1) + bv_0 \geq 0) \vee$
$(w(t) + 2av_0x(t) + bv_0 \leq 0 \wedge w(t_1) + 2av_0x(t_1) + bv_0 \leq 0)) \wedge$
$((w(t_1) + 2av_0x(t_1) + bv_0 \geq 0 \wedge w(t') + 2av_0x(t') + bv_0 \geq 0) \vee$
$(w(t_1) + 2av_0x(t_1) + bv_0 \leq 0 \wedge w(t') + 2av_0x(t') + bv_0 \leq 0))$

4.2.2 Application to non-linear hybrid automata

In the general case of non-linear hybrid automata (here meant as hybrid systems with non-polynomial functions), the reduction of Theorem 3 may result in more complex quantified formulas. Even if we restrict to polynomial invariants, their composition with transcendental primitive solutions may yield complex derivatives. However, in many cases, we can convert the derived quantified formula into a polynomial which is simpler than the original[1].

Example 3 Let us consider a temperature controller. The system is parameterized by the lower and upper temperature limits m and M, the outside temperature u, the rate b of temperature exchanged with the outside, and the rate c of temperature increase due to the heater. The constraints on the parameters are $u < m < M \wedge c > 0 \wedge b > 0$. The HS is defined as follows:

- $V = \{h\}$ where h is a variable representing the heater.
- $X = \{x\}$ where x represents the temperature.
- $Init := m \leq x \leq M$.
- $Inv := (h = 0 \rightarrow x \geq m) \wedge (h = c \rightarrow x \leq M)$.
- $Trans := (h = 0 \rightarrow (x = m \wedge h' = c)) \wedge (h = c \rightarrow (x = M \wedge h' = 0)) \wedge x' = x$.
- $Flow := \dot{x} = b(u - x) + h$.

The primitive solution of the ODE when the location is $h = c$ is $x(t) := u + \frac{(x(0)-u)}{b}e^{(-b*t)} + \frac{c}{b}$. Its derivative is $x(t) := -(x(0) - u)e^{(-b*t)}$, which never changes sign. Therefore, applying Theorem 3, $\forall \epsilon \in [t, t'], x \geq m$ is translated into the formula $x(t) \geq m \wedge x(t') \geq m$ and similarly for $\forall \epsilon \in [t, t'], x \leq M$.

[1] This conversion is not currently automated.

Example 4 Consider the roundabout collision avoidance system example (cfr. e.g. [35]). The continuous dynamics of a safe circular maneuver is described by the following equations $\dot{x_1} = d_1, \dot{x_2} = d_2, \dot{d_1} = -\omega d_2, \dot{d_2} = \omega d_1, \dot{y_1} = e_1, \dot{y_2} = e_2, \dot{e_1} = -\rho e_2, \dot{e_2} = \rho e_1, (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2$.

The primitive solution of the differential equations is:

$$\begin{aligned} x_1 = \frac{1}{\omega} sin(\theta),\ x_2 = -\frac{1}{\omega} cos(\theta), \\ d_1 = cos(\theta),\ d_2 = sin(\theta),\ \theta = \omega t + t_0, \\ y_1 = \frac{1}{\rho} sin(\xi),\ y_2 = -\frac{1}{\rho} cos(\xi), \\ e_1 = cos(\xi),\ e_2 = sin(\xi),\ \xi = \rho t + t_0 \end{aligned}$$

Substituting the primitive solution into the invariant $(x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2$ we obtain the formula:

$$\frac{1}{\omega^2} + \frac{1}{\rho^2} - \frac{2}{\omega\rho} sin(\theta) sin(\xi) - \frac{2}{\omega\rho} cos(\theta) cos(\xi) \geq p^2.$$

which can be rewritten into: $\phi := \frac{1}{\omega^2} + \frac{1}{\rho^2} - \frac{2}{\omega\rho} cos(\theta - \xi) \geq p^2$.

The standard quantified encoding is $\forall t \in [0, \delta], \phi(t)$. Applying Theorem 3, we obtain the formula:

$$\begin{aligned} \phi(0) \wedge \phi(\delta) \wedge\ (\forall t(-sin(\theta - \xi)(\omega - \rho) \geq 0) \vee \\ \forall t(-sin(\theta - \xi)(\omega - \rho) \leq 0)). \end{aligned}$$

The quantified sub-formulas can be rewritten into polynomials over θ and ξ. For example, $\forall t(-sin(\theta - \xi)(\omega - \rho) \geq 0)$ can be rewritten into $\forall t(\omega - \xi \geq 0 \wedge (\pi \leq \theta - \rho \leq 2\pi) \vee \omega - \xi \leq 0 \wedge (0 \leq \theta - \rho \leq \pi))$. Since θ and ρ are linear, this can be converted into an equivalent quantifier-free one.

Example 5 Consider the steering car example of [27]. The flow invariant of the location *correct_left* is $\dot{p} = -r * sin(\gamma), \dot{\gamma} = \omega, \dot{c} = -2, -1 \leq p \leq 1, c \geq 0$. Let us make the example more complex (and realistic) considering an (uniformly) accelerated rotation by adding $\dot{\omega} = \alpha, 0 \leq \alpha \leq 3$, where α is a (real valued) discrete variable.

The semantics of this flow invariant is that during a timed transition of δ time units starting from the state $p(0) = p_0, \gamma(0) = \gamma_0, c(0) = c_0, \omega(0) = \omega_0$, there exist the continuous differentiable functions p, γ, c, ω that satisfy the ODEs $\dot{p} = -r * sin(\gamma), \dot{\gamma} = \omega, \dot{c} = -2, \dot{\omega} = \alpha$ and such that $\forall t, 0 \leq t \leq \delta(-1 \leq p(t) \leq 1 \wedge c(t) \geq 0 \wedge 0 \leq \alpha \leq 3)$.

The removal of quantifiers is very similar to the Example 4.

The quantification can be distributed obtaining:

$$\begin{aligned} I_1 &:= \forall t, 0 \leq t \leq \delta(-1 \leq p(t) \leq 1) \\ I_2 &:= \forall t, 0 \leq t \leq \delta(c(t) \geq 0) \\ I_3 &:= \forall t, 0 \leq t \leq \delta(0 \leq \alpha \leq 3) \end{aligned}$$

I_3 is equivalent to $0 \leq \alpha_0 \leq 3$ since α does not change during the timed transition.

I_2 is equivalent to $c(0) \geq 0$ and $c(\delta) \geq 0$ since c is linear. This can be obtained also from Theorem 3 by replacing $\dot{c}$ with -2 and simplifying.

In order to remove the quantification of I_1, we apply the Theorem 3 by obtaining

$$I_1 \equiv -1 \leq p(0) \leq 1 \wedge -1 \leq p(\delta) \leq 1 \wedge \\ (\forall t, 0 \leq t \leq \delta(\dot{p} \geq 0) \vee \forall t, 0 \leq t \leq \delta(\dot{p} \leq 0))$$

By replacing $\dot{p}$ with $-r * sin\gamma$ we obtain the invariant condition:

$$I_{1a} := \forall t, 0 \leq t \leq \delta(-r * sin(\gamma) \geq 0)$$

This can be solved considering $0 \leq \gamma \leq 2\pi$ by taking $\pi \leq \gamma \leq 2\pi$. This results in the invariant condition:

$$I_{1b} := \forall t, 0 \leq t \leq \delta(\pi \leq \gamma(t) \leq 2\pi)$$

Applying again Theorem 3, we obtain an equivalent formula containing $\forall t, 0 \leq t \leq \delta(\dot{\gamma} \geq 0)$. Now, since γ is linear we can remove the quantification in the standard way.

5 Encoding HS with Polynomial Dynamics into Transition Systems

In this section, we show how Theorem 3 can be exploited to automatically encode a HS with polynomial dynamics into a transition system with quantifier-free formulas.

Theorem 3 states the existence of the points $t_1, \ldots, t_n$ where the derivative changes sign. However, such points are unknown. The encoding of a HS into a transition system must thus implicitly represent when the derivative of the invariant changes sign. This is achieved by simply forcing that the sign of the derivative is constant throughout the timed transition. The encoding implicitly concatenates timed transitions one after the other, delegating to the search the task of finding the sequence of time points that split the interval, so that the sign of the derivative is uniformly constant in the resulting trace.

Given a formula T including the invariant condition $\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0$, the condition can be locally replaced with $g(t) \bowtie 0 \wedge g(t') \bowtie 0 \wedge \mathit{Constant}(\dot{g}, t, t')$ obtaining a new formula $\tau(T)$.

τ performs a recursive substitution of the quantified expressions. The recursion terminates when the quantified formula is a linear polynomial, thus allowing to the quantifiers to be trivially removed. τ is defined recursively as follows:

$$\begin{aligned} \tau(\psi_1 \wedge \psi_2) &:= \tau(\psi_1) \wedge \tau(\psi_2) \\ \tau(\psi_1 \vee \psi_2) &:= \tau(\psi_1) \vee \tau(\psi_2) \\ \tau(\neg\psi) &:= \neg\psi, \ (\psi \text{ is a predicate}) \\ \tau(\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0) &:= \begin{cases} g(t) \bowtie 0 \wedge g(t') \bowtie 0 & \text{if } g \text{ linear} \\ g(t) \bowtie 0 \wedge g(t') \bowtie 0 \wedge & \\ \tau(\mathit{Constant}(\dot{g}, t, t')) & \text{otherwise} \end{cases} \end{aligned} \tag{1}$$

The correctness of the transformation is given by the following theorem.

Theorem 6 *If S_D is the encoding of the HS S and $\tau(S_D)$ is the transition system obtained by replacing $Trans$ with $\tau(Trans)$, then $\tau(S_D)$ is the encoding of a sampling refinement of S.*

Proof ($\Leftarrow$) If a sequence of states satisfies $\tau(S_D)$, then by Theorem 3, the sequence satisfies also S_D, and by Theorem 1, it represents a path of S. ($\Rightarrow$) Consider a hybrid trace $\langle f_0, I_0\rangle, \langle f_1, I_1\rangle, \ldots, \langle f_k, I_k\rangle$ which is a path of S. Assuming that $\dot{g}$ has finite variability, we can refine the hybrid trace into a new hybrid trace in which $\dot{g}$ is constant in every interval. The new hybrid trace also satisfies S by Theorem 2 and thus the corresponding discrete trace $s_0, \ldots, s_k$ satisfies its encoding S_D. At every i, if s_i satisfies $\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0$, then both $f(s_i, t)$ and $f(s_i, t')$ satisfy $g \bowtie 0$. Since $\dot{g}$ has constant sign in I_i, s_i satisfies also $\tau(Trans)$. Therefore the discrete trace satisfies also $\tau(S_D)$.

The recursive definition of τ in (1) creates a formula whose size is exponential in the degree of the polynomial inside the invariant. We use the following equivalence to keep the size of the encoding *linear* in the degree of the polynomial (here g is not linear):

$$\begin{aligned}\tau(\mathit{Constant}(g, t, t')) &= (g(t) \geq 0 \wedge g(t') \geq 0 \wedge \tau(\mathit{Constant}(\dot{g}, t, t'))) \vee \\ &\quad (g(t) \leq 0 \wedge g(t') \leq 0 \wedge \tau(\mathit{Constant}(\dot{g}, t, t'))) \\ &= ((g(t) \geq 0 \wedge g(t') \geq 0) \vee (g(t) \leq 0 \wedge g(t') \leq 0)) \wedge \\ &\quad \tau(\mathit{Constant}(\dot{g}, t, t'))\end{aligned}$$

The sequential encoding may force the split of a continuous transition in several transitions, since the predicates introduced to remove the quantifiers force the derivatives of the invariant conditions to be constant. While the encoding enables to remove the quantifier, the depth of the bounded model checking formula may increase due to the splitting. In incremental bounded model checking, the burden of finding how many splits are necessary is delegated to the search.

In the case of polynomial hybrid automata we can compute an upper bound on the number of consecutive continuous transitions (continuous transitions not separated by a discrete transition) needed to simulate the longest quantified continuous transition (the continuous transition with the maximum time elapse).

We can compute the upper bound on the number of intervals needed to "cover" the quantified continuous transition for the invariant predicate $\forall \epsilon \in [t, t'], g(\epsilon) \bowtie 0$. If $\Omega(g)$ is the degree of the polynomial, then the maximum number of intervals that have to be considered is $ub(g) = \frac{\Omega(g) * (\Omega(g)-1)}{2}$. In fact, the i-th derivative of g has degree $\Omega(g) - i$ and thus changes sign $\Omega(g) - i$ times.

6 Encoding of Linear Hybrid Systems into Transition Systems

In this section we describe how we can encode subclasses of *Linear Hybrid Systems* using quantifier free formulas. We rely on the results presented by [30], which show how the primitive solution of several classes of linear hybrid systems can be encoded in the thory of reals. While their approach trivially handles invariants, it still relies on universal quantification. We show that in two cases the universal quantifier can be removed by applying Theorem 5. The process is straightforward in one case, while it requires several steps in the other.

We consider *Linear Hybrid Systems* where each flow condition is of the form $\dot{X} = AX + b$, where $A \in \mathbb{R}^{n \times n}, b : \mathbb{R}^n \to \mathbb{R}^n$ [2].

Given a matrix $M \in \mathbb{R}^q \times \mathbb{R}^s$, with $q, s \in \mathbb{N}$, we will write M_{ij} to refer to the element at the i-th row and at the j-th column of M. We will avoid one index in the case of vectors, where $s = 1$. Also, we denote with Λ the set of eigenvalues of A. To ease the presentation, we will use the symbol δ to represent the amount of time $t' - t$ elapsed in a continuous transition.

Let L be a set of symbolic parameters that can be used in the inputs b. The value of each $l \in L$ is unknown, but it does not change during the execution of the system. Moreover, the set L is disjoint from the set of the variables of the hybrid system $(X \cup V)$. Let P be a set of functions over δ (e.g. $P = \{p(\delta) = \delta^n, n \in \mathbb{N}\}$ is the set of all the powers of δ with a natural exponent). Let M_P be a set of inputs paramterized by P:

$$M_P := \left\{ b \in [b_1, \ldots, b_n]^T \mid \text{for } i \in [1, n], b_i(\delta) = \sum_{l=1}^{r} u_{i,l} p_l(\delta), \right. \\ \left. p_l(\delta) \in P,\ u_{i,l} \text{ is a } \Sigma_{\mathbb{R}}\text{-formula over } L \right\}$$

P determines the function of the terms $p_l(\delta)$. We will consider different families of inputs, parameterized by different families of functions P.

Given a linear system $\dot{X} = AX + b$ the reachability problem can be expressed in the theory of reals if the matrix A and the inputs b have a particular structure[30]:

- A is nilpotent and $P = \{\delta^n, n \in \mathbb{Z}\}$.
- A is diagonalizable, all its eigenvalues are real and $P = \{e^{u_l \delta}, u_l \notin \Lambda, u_l \in \mathbb{Q}\}$.
- A is diagonalizable, all its eigenvalues are imaginary and $P = \{\sin(u_l \delta), u_l \notin \Lambda, u_l \in \mathbb{Q}\} \cup \{\cos(u_l \delta), u_l \notin \Lambda, u_l \in \mathbb{Q}\}$.

We will provide a quantifier-free encoding for the first two cases.

Note that we do not consider symbolic coefficients in the matrix A. While in the first case obtaining an exact solution in the theory of reals is straightforward, also in the presence of symbolic coefficients of the matrix, the second case is more involved and, since its applicability depends on the coefficient of A (i.e. eigenvalues and diagonalizability), it requires imposing several constraints on the parameters.

The general solution of $\dot{X} = AX + b$ is:

$$X(\delta) = f(X, \delta, b) = e^{A\delta} X + \Psi(X, \delta, b)$$

where:

$$\Psi(X, \delta, b) = \int_{s=0}^{\delta} e^{A(\delta - s)} b(s)\, \mathrm{d}s \qquad e^{A\delta} = \sum_{k=0}^{\infty} \frac{\delta^k}{k!} A^k$$

6.1 Reduction in the nilpotent case

Suppose that the flow condition in the location of a hybrid automaton is of the form $\dot{X} = AX + b$, where A is nilpotent and the inputs are of the form $P = \{\delta^i, i \in \mathbb{N}\}$.

[2] This definition is sufficient to represent input BU where $B \in \mathbb{R}^{n \times m}$ and $U : \mathbb{R}^m \to \mathbb{R}^n$

Also, suppose that $g(X,\delta) \bowtie 0$ be the invariant in the location, and $g(X,\delta)$ be a multivariate polynomial with variables in X and $P = \{\delta^i, i \in \mathbb{N}\}$.

In this case, for each $1 \leq i \leq n$ the solution is:

$$f(X,\delta,b)_i := \sum_{k=1}^{n} \gamma_{ij}(X) + \delta^{k-1} + \sum_{k=0}^{v} \rho_{i,k}(b)\delta^{k-1}$$

for some $v \in \mathbb{N}$, some polynomials $\rho_{i,k}(b)$ and $\gamma_{ij}(X) = \sum_{j=1}^{n} (A_k)_{ij} x_j \frac{1}{k!}$. Note that we do not know a priori $\sum_{k=0}^{v} \rho_{i,k}(b)\delta^k$, since it depends on the inputs b. However, an expression of this form can always be obtained as a solution of the integral in $\Psi(X,\delta,b)$ (i.e. in this case we are just integrating a polynomial over t). Thus, the primitive solution $f(X,\delta,b)$ is a $\Sigma_{\mathbb{R}}$-formula without transcendental functions (i.e. a polynomial).

Also the invariant $g(X,\delta)$ is a multivariate polynomial over X and $P = \{\delta^i, i \in \mathbb{N}\}$. The encoding of the continuous transition in the location is:

$$\delta > 0 \wedge \bigwedge_{i}^{n} f(X,\delta,b)_i \wedge \forall \epsilon \in [0,\delta], g(X,\epsilon) \bowtie 0$$

We are in the case of polynomial hybrid automata, so we apply the Theorem 5 to obtain a quantifier-free encoding.

Note that the bouncing ball example (see Example 2) with a set of instantiated parameters, falls in this class of systems.

Example 6 The movement of two vehicles which follows a uniformly accelerated motion can be modeled with the linear dynamic $\dot{X} = AX + b$ where: $X^T := \begin{bmatrix} x_1 & v_1 & a_1 & x_2 & v_2 & a_2 \end{bmatrix}$. $A := \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$ and $b^T := \begin{bmatrix} 0 & 0 & 10 & 0 & 0 & 6 \end{bmatrix}$. The invariant of the system $x_1 \leq x_2 + s_d$ guarantees that the first vehicle follows the second vehicle respecting a safety distance s_d. The primitive solutions are: $x_1(\delta) = 5t^2 + v_1(0)t + x_1(0)$, $v_1(\delta) = 10t + v_1(0)$, $a_1(\delta) = 10$, $x_2(\delta) = 3t^2 + v_2(0) + x_2(0)$, $v_2(\delta) = 6t + v_2(0)$, $a_2(\delta) = 6$. The quantified invariant may be rewritten as:

$$\forall \epsilon \in [0,\delta], 2\epsilon^2 + (v_1(0) - v_2(0))\epsilon + x_1(0) - x_2(0) - s_d \leq 0$$

Thus, applying Theorem 5 we remove the quantifers:

$$\begin{aligned} x_1(0) - x_2(0) - s_d \leq 0 \wedge \\ 2\delta^2 + (v_1(0) - v_2(0))\delta + x_1(0) - x_2(0) - s_d \leq 0 \wedge \\ (v_1(0) + v_1(0) \geq 0 \wedge 4 * \delta + v_1(0) + v_1(0) \geq 0) \wedge \\ (v_1(0) + v_1(0) < 0 \wedge 4 * \delta + v_1(0) + v_1(0) < 0) \end{aligned}$$

6.2 Reduction in the case A is diagonalizable with real eigenvalues

Suppose that the flow condition in the location of a hybrid automaton is of the form $\dot{X} = AX + b$, where A is diagonalizable, all its eigenvalues are real (i.e. $\Lambda \subset \mathbb{R}$) and the inputs are of the form $P = \{e^{u_l \delta}, u_l \notin \Lambda, u_l \in \mathbb{Q}\}$. Also, suppose that $g(X, \delta) \bowtie 0$ is the invariant in the location, and $g(X, \delta)$ is a multivariate polynomial with variables in X and $P = \{e^{u_l \delta}, u_l \notin \Lambda, u_l \in \mathbb{Q}\}$ [3]. Since A is diagonalizable, there exists an invertible matrix $T \in \mathbb{R}^{n \times n}$ such that $A = TDT^{-1}$, where $D = \begin{bmatrix} \lambda_1 & & \\ & \dots & \\ & & \lambda_n \end{bmatrix}$ is the diagonal matrix of A. Hence, we can compute $e^{A\delta} = e^{TDT^{-1}\delta} = T \begin{bmatrix} e^{\lambda_1 \delta} & & \\ & \dots & \\ & & e^{\lambda_n \delta} \end{bmatrix} T^{-1}$.

In this case, for each $1 \leq i \leq n$ we have:

$$f(X, \delta, b)_i := \sum_{k=1}^{n} \gamma_{i,k}(x) e^{\lambda_k \delta} + \sum_{k=1}^{s} \psi_{ik}(b) e^{\nu_k \delta}$$

where $s \in \mathbb{N}$, $\gamma_{i,k}(x)$ is a polynomial, and for all $1 \leq k \leq s$, $\nu_k \in \mathbb{Q}$ and $\psi_{ik}(b)$ is a polynomial. Let us write $f(X, \delta, b)_i$ as follows (for some natural $q > 0$):

$$f(X, \delta, b)_i := \sum_{k=1}^{q} \phi_{ik}(X, b) e^{\eta_k \delta}$$

The formulas $f(X, \delta, b)_i$ are such that δ occurs only in the exponent of e, and hence we have terms like $e^{\eta\delta}$, where $\eta \in \mathbb{Q}$. We also required the same property in the invariant g. Thus, we substitute the exponential with a new variable z both in the solution and in the invariant as follows:

- We compute a common denominator d among all the rational coefficients η (i.e. $d = \prod_{\textit{for all the coefficients } \eta} den(\eta)$), where $den(\eta)$ is the denominator of η.
- We define the variable $z = e^{\frac{\delta}{d}}$.
- We substitute $e^{\frac{\delta}{d}}$ with z in the solution and in the invariant:
  - for all $1 \leq i \leq n$ $f(X, z, b)_i := \hat{f}(X, \delta, b)[z/e^{\frac{\delta}{d}}]_i)$.
  - $g(\hat{X}, z) := g(X, \delta)[z/e^{\frac{\delta}{d}}]$.
- If $\hat{f}$ contains a negative power of z, i.e., some z^{-l} with $l > 0$, we substitute it with w^l, where $zw = 1$:
  - $1 \leq i \leq n$ $\hat{f}(X, z, w, b) = \hat{f}(X, z, b)[w^l/z^{-l}]_i \wedge wz = 1$.

Remark 6 If we want to obtain a polynomial, the substitution of $e^{\frac{\delta}{d}}$ with z forbids the use of the variable t, which tracks the total amount of elapsed time, in the transition system. In fact, t is updated in the continuous transition using a logarithm ($t' = \ln(z) - t$). This shows that in the system we cannot have continuous variables that evolve as clocks (i.e. a variable x such that $\dot{x} = 1$).

[3] Note that u_l cannot be an eigenvalue of the system. This condition is necessary to get a solution where δ appears only as exponent of e, thus enabling the removal of the exponential function via substitution.

The encoding of the continuous transition in the location is:

$$\delta > 0 \wedge \bigwedge_{i}^{n} f(X, \delta, b)_i \wedge \forall \epsilon \in [0, \delta], g(\epsilon) \bowtie 0$$

Performing the substitution we obtain the following formula:

$$z > 1 \wedge zw = 1 \wedge \bigwedge_{i}^{n} \hat{f}(X, z, w, b)_i \wedge \forall z_\epsilon \in [1, z], \hat{g}(X, z_\epsilon) \bowtie 0$$

Note that $\hat{g}(X, z_\epsilon)$ may contain negative powers of z. However, $z > 0$ and we can multiply both sides of $\hat{g}(X, z_\epsilon)$ by z^l where l is the greatest negative order with which occurs in $\hat{g}$. Now we can recursively apply Theorem 3 to obtain a quantifier free formula.

Example 7 Consider a system with $X^T = [x, y]$, $A = \begin{bmatrix} 1 & \frac{-12}{5} \\ \frac{-12}{5} & 1 \end{bmatrix}$, $b^T = [1, 1]$ and invariant $x^2 \geq 0$. The eigenvalues of A are $\frac{-7}{5}$ and $\frac{17}{5}$ and A is diagonalizable. The solution $X(\delta)$ is:

$$x(\delta) := \frac{x-y}{2} e^{\frac{17}{5}\delta} + \frac{x+y}{2} e^{-\frac{7}{5}\delta} - \frac{5}{7} e^{-\frac{7}{5}\delta} + \frac{5}{7}$$
$$y(\delta) := \frac{y-x}{2} e^{\frac{17}{5}\delta} + \frac{x+y}{2} e^{-\frac{7}{5}\delta} - \frac{5}{7} e^{-\frac{7}{5}\delta} + \frac{5}{7}$$

In this case we use the following variables for the substitution (note that the $d = 5$): $z = e^{\frac{1}{5}\delta}$:

$$\hat{x}(z) := \frac{x-y}{2} z^{17} + \frac{x+y}{2} z^{-7} - \frac{5}{7} z^{-7} + \frac{5}{7}$$
$$\hat{y}(z) := \frac{y-x}{2} z^{17} + \frac{x+y}{2} z^{-7} - \frac{5}{7} z^{-7} + \frac{5}{7}$$

$\hat{g}(z)$ is:

$$\hat{g}(z) := (\frac{5}{7}x + \frac{5}{7}y + \frac{x+y}{2} + \frac{25}{49}) z^{-14} + (\frac{5}{7}x + \frac{5}{7}y + \frac{50}{49}) z^{-7} +$$
$$(\frac{x^2}{2} + \frac{5}{7}x - \frac{y^2}{2} - \frac{5}{7}y) z^{-7} z^{17} + \frac{x-y}{2}^2 z^{34} (\frac{5}{7}x - \frac{5}{7}y) z^{17} + \frac{25}{49}$$

Then, we multiply both sides of $\hat{g}(z) \bowtie 0$ by z^{14}:

$$\frac{x-y}{2}^2 z^{48} + (\frac{5}{7}x - \frac{5}{7}y) z^{31} + (\frac{x^2}{2} + \frac{5}{7}x - \frac{y^2}{2} - \frac{5}{7}y) z^{24} + \frac{25}{49} z^{14}$$
$$(\frac{5}{7}x + \frac{5}{7}y + \frac{50}{49}) z^7 + (\frac{5}{7}x + \frac{5}{7}y + \frac{x+y}{2} + \frac{25}{49}) \bowtie 0$$

7 Related work

The quantifier-free encoding that we propose is related to quantifier elimination procedures (see, e.g., [19]). It is not a quantifier elimination procedure in that it contains new variables that are implicitly existentially quantified. In fact, we apply the reduction even in some cases of transcendental functions. The burden to remove the quantifiers is delegated to the verification techniques if necessary. We claim that quantifier elimination is somehow an overkill: the verification techniques does not often need the precise region of points where the invariant holds; it is usually sufficient either to pick some "good" values (in case of reachability) or to find "good" invariants (in case of safety verification).

Several works focus on the reachability problem for hybrid systems, but they use less expressive invariants or they restrict the class of the analyzed hybrid automata. We extend the bounded model checking encoding of linear hybrid automata [5, 1], where invariants hold iff they hold at the first and the last instant of a timed transition, thus the resulting encoding is quantifier free. Other approaches [11, 21, 27] focus on non-linear hybrid automata. In [11], the authors solve the reachability problem for non-linear convex hybrid automata. The restriction to convex invariant and linear flow conditions, or to monotonic invariant and convex flow, allows an easy encoding of invariants without quantifiers. Many examples, including those mentioned in this paper, do not fall in this class of automata. In [21] the authors propose an SMT solver modulo ODEs, that can be used to perform bounded model checking on hybrid automata where the flow conditions are ODEs. The only allowed invariants are of the form $x \in [l, u]$, where x is a continuous variable and $l, u \in \mathbb{R}$. Their main focus is on the integration of numerical methods to compute the initial value problem for ODEs, while they cannot manage more complex invariants (e.g. linear functions). ODEs are also handled directly in [27]. This is done by computing the precise intersection of the continuous flow with the guards of the hybrid automaton. The solver can in principle handle invariants, but the authors state that the implementation is not mature enough to evaluate the approach. Approaches based on motion planning [33] do not encode symbolically the invariants, since they simulate the ODEs using numerical methods. In contrast, we encode a set of continuous transitions.

The prominent approaches to the verification of HSs are based either on the exploration of the reachable states or on deductive systems. We refer the readers to [2] for a recent survey. The focus of our work is on the SMT-based paradigm, which, although less mature, seems promising.

Our settings also differs from the works that build abstractions for HSs. The approaches described in [39, 37] use techniques based on the sign of derivatives such as ours. However, the purpose is different in that they generate over-approximations of the HS.

Finally, we mention the "clock translation" described in [25], where invariants are translated into constraints on time. However, the translation is restricted to monotonic flows (plus other restrictions on the independence of variables).

8 Experimental evaluation

8.1 Benchmarks

We applied our approach to several benchmarks of non-linear hybrid automata, obtaining a quantifier-free encoding. For the polynomial subcase we used the ETCS benchmark [26], an industrial case study of the braking control system of trains, the classic bouncing ball, and a simple ballistics example. For the bouncing ball, we used four variants: a ball moving vertically in one dimension and bouncing on a plain floor, a two-dimensional variant with constant horizontal speed, a third variant still in two-dimensions but bouncing on a hill (vertical parabola), and a fourth variant bouncing on a slope (horizontal parabola). As for the ballistics example, we modeled an object that flies above an obstacle keeping below a certain ceiling. As for nonlinear benchmarks with transcendental functions, we used the temperature controller, the roundabout collision avoidance system and the steering car mentioned in Sec. 4.2.2. All the benchmarks are publicly available at `http://es.fbk.eu/people/mover/tests/FMSD_FMCADSI/`.

8.2 Implementation and optimizations

The techniques discussed in the previous sections have been implemented in an extension of NuSMV[4], which is able to deal with HSs formalized in the HyDI language [14]. The NuSMV extension features an SMT-based approach to the verification of HSs, including bounded model checking and inductive reasoning. We automatically encode the invariants for polynomial hybrid automata, while we manually encode the invariants for the other benchmarks. iSAT[5] is used as the backend to solve the resulting satisfiability queries.

Another optimization that we implemented is the use of some *lemmas* that relate the value of polynomials to the value of their derivatives. More specifically, we optionally add to τ the following formulas:

$$(\dot{g}(t) > 0 \vee \dot{g}(t') > 0) \rightarrow ((g(t) \geq 0 \rightarrow g(t') \geq 0) \wedge (g(t') \leq 0 \rightarrow g(t) \leq 0)) \wedge$$
$$(\dot{g}(t) < 0 \vee \dot{g}(t') < 0) \rightarrow ((g(t) \leq 0 \rightarrow g(t') \leq 0) \wedge (g(t') \geq 0 \rightarrow g(t) \geq 0))$$

The formula means that when the derivative is positive g can only increase (thus cannot pass from positive to negative) and vice versa when $\dot{g}$ is negative g can only decrease (thus cannot pass from negative to positive).

In the BMC settings we usually perform a search where we check if the target is violated for an increasing path length. In principle, the removal of the quantifiers requires more continuous transitions, thus increasing the size of the formula passed to the solver. It is convenient to use a "layered" approach, where we first reach the target in an over-approximation of the HS, where invariants are not guaranteed to hold, and then we check if there exists a path that reaches the target and for which invariants hold.

[4] http://nusmv.fbk.eu/
[5] http://isat.gforge.avacs.org/

	# vars	Max degree	RedLog	QepCAD
etcs_braking	4	2	0.14	0.05
ball_1d_plain	4	2	0.10	0.03
ball_2d_plain	4	2	0.10	0.03
ball_2d_hill	5	2	0.15	T.O. > 3600.00
ball_2d_slope	5	4	N.A.	T.O. > 3600.00
simple_ballistics	5	4	N.A.	T.O. > 3600.00

Table 1 Results of applying quantifier elimination to the polynomial benchmarks (max degree is the maximum degree of the quantified variable,T.O.is a time out of 3600 seconds, N.A. means not applicable).

8.3 Experimental setting and results

We evaluated the alternative use of quantifier elimination procedures, within their range of applicability, i.e. polynomial hybrid automata. We experimented with Cylindrical Algebraic Decomposition (CAD) (using QepCAD[6]) and Virtual Substitution (using RedLog[7]). Table 1 reports, for each polynomial benchmark, the time needed to obtain a quantifier free formula of the invariants using QepCAD and RedLog. The Virtual Substitution approach of RedLog can only handle formulas quantified over a quadratic variable. QepCAD is slightly more general, but de facto less useful: the results highlight the dramatic computational complexity of the procedure (e.g. *ball_2d_hill*, with 5 variables, times out in one hour). Thus, the quantifier elimination approach cannot even handle the polynomial benchmarks ballistic and *ball_2d_slope* (in addition to the benchmarks with transcendental functions).

We used the bounded model checking functionalities enabled by our approach to validate the various models and to evaluate the performance of the invariant encoding. For each model we generated different reachability properties which are falsified by traces with an increasing length. We evaluated the encoding of the invariant by comparing the time needed to find these traces with BMC. When quantifier elimination was able to produce a result, we also compared it with our approach using the same SMT-based technique, in order to evaluate the overhead caused by the splitting. The results are shown in Table 2. The encoding time of our approach is instantaneous in all cases. In the cases where quantifier elimination is feasible, the resulting encoding may induce traces with a smaller number of steps, because timed transitions must not be split. This happens for the *ball_1d_plain* and the *ball_2d_hill* benchmarks. The reduced number of steps also reduces the time needed to generate the trace.

Our approach was also able to prove a simple invariant on the ballistics example, that was beyond the applicability of SMT-based techniques. We chose as obstacle a circle shape with center in $(c, 0)$ and radius r. If the ceiling level is less than r, the object cannot clearly pass. This has been proved with NuSMV and iSAT. Ignoring the invariant along the timed transitions (keeping it only on the discrete points) allows for spurious traces that forbid the inductive proof. Note that this small example is beyond the applicability of quantifier elimination (see Table 1).

[6] http://www.usna.edu/cs/ qepcad/B/QEPCAD.html
[7] http://redlog.dolzmann.de/

	quantifier-free encoding	qelim (qepcad)	qelim (redlog)
etcs_braking	66.75 / 17	161.52 / 17	168.16 / 17
ball_1d_plain.01	0.05 / 2	0.05 / 2	0.03 / 2
ball_1d_plain.02	25.50 / 6	0.09 / 4	0.06 / 4
ball_1d_plain.03	31.43 / 10	0.28 / 6	0.40 / 6
ball_1d_plain.04	36.23 / 14	0.46 / 8	0.65 / 8
ball_1d_plain.05	151.41 / 18	1.27 / 10	1.51 / 10
ball_2d_plain.01	0.08 / 2	0.18 / 2	0.28 / 2
ball_2d_plain.02	4.20 / 6	3.14 / 6	3.64 / 6
ball_2d_plain.03	16.04 / 10	15.90 / 10	62.64 / 10
ball_2d_hill.01	1.30 / 4	na / na	0.94 / 2
ball_2d_hill.02	118.67 / 8	na / na	15.36 / 4
ball_2d_slope.01	to / na	na / na	na / na
simple_ballistics	8.31 / 1	na / na	na / na

Table 2 Results (running time / path length) of BMC with the different encodings.

Some remarks are in order. Our approach strongly depends on the availability of SMT solvers for quantifier-free theories of nonlinear arithmetic, to solve the formulas resulting from our SMT-based verification engines. To this end, we tried to use all the available solvers for nonlinear arithmetic: Z3[8], SMT-RAT[9], CVC3[10], miniSMT[11], RAHD[12], hydlogic[13], dReal[14]. and iSAT[15]. Z3 and SMT-RAT implement two complete decision procedures for the non-linear arithmetic over reals. currently, neither solver integrates a layering with the linear arithmetic solver: in this case all the linear arithmetic constraints are handled using the non-linear solver, thus resulting in an inefficient approach. This is the case for our BMC case studies, which have a significant part of linear constraints. Instead, CVC3 and miniSMT implement an incomplete decision procedure for non-linear arithmetic (and miniSMT is tailored only to check satisfiable formulas). As a result, these solvers turned out to return "unknown" on most of the queries generated from our benchmarks. The hydlogic system turned out to be immature, while RAHD exports functionalities that are closer to a theory solver than a full SMT solver, requiring an explicit treatment of disjunctions. iSAT and dReal differ from the other solvers, since they can also provide non-precise solutions. dReal returns an unsatisfiable answer or a satisfiable answer if the formula is satisfiable under a bounded numerical perturbations. iSAT may return "unknown" exposing the results of interval constraints propagation: it produces the intervals found in the search, if these are below a user-defined threshold, as a candidate solution. In many practical cases, this is not spurious, and represents a satisfying assignment of the formula.

[8] http://research.microsoft.com/en-us/um/redmond/projects/z3/
[9] http://smtrat.sourceforge.net/
[10] http://cs.nyu.edu/acsys/cvc3/
[11] http://cl-informatik.uibk.ac.at/software/minismt/
[12] http://homepages.inf.ed.ac.uk/s0793114/rahd/
[13] http://code.google.com/p/hydlogic/
[14] http://www.cs.cmu.edu/ sicung/dReal/
[15] http://isat.gforge.avacs.org/

Overall, despite some recent progress, our experience has shown that the field still requires additional research to deliver what our approach requires, both in terms of completeness, and performance. However, we argue that our method is valuable regardless of the current status of SMT for nonlinear arithmetic. First, we proposed a solution to a problem that was a show-stopper for SMT-based verification. In fact, we are now able to solve some benchmarks that cannot be solved by overapproximation, just forgetting about the quantified invariants. Second, we are hopeful that the field of SMT can deliver quick progress in quantifier-free nonlinear arithmetic. In fact, the development of SMT solving for non-linear arithmetic has been influenced by benchmarks from other domains (e.g. most of the SMT-LIB benchmarks in NRA are from the software domain). To this extent, we generated and submitted to the SMT-LIB a vast number of benchmarks, that will trigger additional research in practically relevant directions.

9 Conclusions

In this paper, we tackled the problem of dealing with invariant constraints in hybrid systems in the setting of SMT-based verification. This is largely an open problem, due to the presence of the universal quantifiers required to encode that the invariant must hold throughout all time instants in delay transitions.

We proposed new methods that allow for the reduction to quantifier-free theories, at the cost of introducing additional variables. Our approach provides a comprehensive handling of invariants. First, it allows us to handle a wide range of hybrid systems, either automatically, as in the case of polynomial hybrid automata and for some subclasses of linear hybrid systems, or manually, since we may apply several patterns of reduction to obtain quantifier-free formulas. Then, the approach provides a uniform technique to encode the invariants, since it can be applied also in the presence of disjunctions, which were not considered in the existing encodings. As a result, we extend the applicability of the SMT-based verification methods, and were able to verify some novel benchmark problems.

The approach opens several lines of research. First, we will experiment with different layered approaches to the analysis of non-linear constraints, where less expensive (e.g. linear) solvers are applied whenever possible before resorting to expensive but more precise nonlinear solvers. Then, we will use the encoding to simulate the abstract paths obtained by an abstraction-refinement framework (we may consider different abstractions, for example [37, 32]). This use case may slightly differ from the BMC scenario since the search performed by the SMT solver may be partially constrained by the abstract path. Then, we will generalize the approach to the analysis of networks of hybrid automata; in particular, we will exploit the locality of the splits of the continuous transitions in the local time semantics framework. Finally, we will apply the proposed techniques to the analysis of requirements expressed in HRELTL logic [17]. In fact, HRELTL requires the predicates to be constant in arbitrary intervals of time.

References

1. E. Ábrahám, B. Becker, F. Klaedtke, and M. Steffen. Optimizing Bounded Model Checking for Linear Hybrid Systems. In *VMCAI*, pages 396–412, 2005.

2. R. Alur. Formal verification of hybrid systems. In *EMSOFT*, pages 273–278, 2011.
3. R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *Hybrid Systems*, pages 209–229, 1992.
4. E. Asarin, T. Dang, O. Maler, and O. Bournez. Approximate Reachability Analysis of Piecewise-Linear Dynamical Systems. In *HSCC*, pages 20–31, 2000.
5. G. Audemard, M. Bozzano, A. Cimatti, and R. Sebastiani. Verifying Industrial Hybrid Systems with MathSAT. *ENTCS*, 119(2):17–32, 2005.
6. C.W. Barrett, R. Sebastiani, S.A. Seshia, and C. Tinelli. Satisfiability Modulo Theories. In *Handbook of Satisfiability*, pages 825–885. 2009.
7. A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu. Symbolic Model Checking without BDDs. In *TACAS*, pages 193–207, 1999.
8. M. Bozzano, A. Cimatti, J.-P. Katoen, V.Y. Nguyen, T. Noll, and M. Roveri. Safety, Dependability and Performance Analysis of Extended AADL Models. *Comput. J.*, 54(5):754–775, 2011.
9. M. Bozzano, A. Cimatti, J.-P. Katoen, V.Y. Nguyen, T. Noll, M. Roveri, and R. Wimmer. A Model Checker for AADL. In *CAV*, pages 562–565, 2010.
10. L. Bu, A. Cimatti, X. Li, S. Mover, and S. Tonetta. Model Checking of Hybrid Systems using Shallow Synchronization. In *FORTE*, 2010.
11. L. Bu, J. Zhao, and X. Li. Path-Oriented Reachability Verification of a Class of Nonlinear Hybrid Automata Using Convex Programming. In *VMCAI*, pages 78–94, 2010.
12. A. Casagrande, K. Casey, R. Falchi, C. Piazza, B. Ruperti, G. Vizzotto, and B. Mishra. Translating Time-Course Gene Expression Profiles into Semi-algebraic Hybrid Automata Via Dimensionality Reduction. In *AB*, pages 51–65, 2007.
13. A. Cimatti, S. Mover, and S. Tonetta. Efficient Scenario Verification for Hybrid Automata. In *CAV*, pages 317–332, 2011.
14. A. Cimatti, S. Mover, and S. Tonetta. HyDI: a language for symbolic hybrid systems with discrete interaction. In *EUROMICRO-SEAA*, 2011.
15. A. Cimatti, S. Mover, and S. Tonetta. Proving and Explaining the Unfeasibility of Message Sequence Charts for Hybrid Systems. In *FMCAD*, 2011.
16. A. Cimatti, S. Mover, and S. Tonetta. A quantifier-free SMT encoding of non-linear hybrid automata. In *FMCAD*, pages 187–195, 2012.
17. A. Cimatti, M. Roveri, and S. Tonetta. Requirements Validation for Hybrid Systems. In *CAV*, pages 188–203, 2009.
18. L. de Alfaro and Z. Manna. Verification in Continuous Time by Discrete Reasoning. In *AMAST*, pages 292–306, 1995.
19. A. Dolzmann, T. Sturm, and V. Weispfenning. Real quantifier elimination in practice. In *Alg. Algebra and Number Theory*, pages 221–247. Springer, 1998.
20. A. Eggers, M. Fränzle, and C. Herde. SAT Modulo ODE: A Direct SAT Approach to Hybrid Systems. In *ATVA*, pages 171–185, 2008.
21. A. Eggers, N. Ramdani, N. Nedialkov, and M. Fränzle. Improving SAT Modulo ODE for Hybrid Systems Analysis by Combining Different Enclosure Methods. In *SEFM*, pages 172–187, 2011.
22. M. Fränzle. What Will Be Eventually True of Polynomial Hybrid Automata? In *TACS*, pages 340–359, 2001.
23. G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. SpaceEx: Scalable Verification of Hybrid Systems. In *CAV*, pages 379–395, 2011.
24. S. Graf and H. Saïdi. Construction of Abstract State Graphs with PVS. In *CAV*, pages 72–83, 1997.
25. T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. Algorithmic Analysis of Nonlinear Hybrid Systems. 1998.
26. C. Herde, A. Eggers, M. Fränzle, and T. Teige. Analysis of Hybrid Systems Using HySAT. In *ICONS*, pages 196–201, 2008.
27. D. Ishii, K. Ueda, and H. Hosobe. An interval-based SAT modulo ODE solver for model checking nonlinear hybrid systems. *STTT*, 13(5):449–461, 2011.
28. S. Jha, B. A. Brady, and S. A. Seshia. Symbolic Reachability Analysis of Lazy Linear Hybrid Automata. In *FORMATS*, pages 241–256, 2007.
29. T. King and C. Barrett. Exploring and Categorizing Error Spaces using BMC and SMT. In *SMT*, 2011.

30. Gerardo Lafferriere, George J. Pappas, and Sergio Yovine. Symbolic reachability computation for families of linear vector fields. *J. Symb. Comput.*, 32(3):231–253, 2001.
31. K.L. McMillan. Interpolation and SAT-Based Model Checking. In *CAV*, pages 1–13, 2003.
32. Sergio Mover, Alessandro Cimatti, Ashish Tiwari, and Stefano Tonetta. Time-aware relational abstractions for hybrid systems. In *EMSOFT*, pages 1–10, 2013.
33. E. Plaku, L.E. Kavraki, and M.Y. Vardi. Hybrid systems: from verification to falsification by combining motion planning and discrete search. *Formal Methods in System Design*, 34(2):157–182, 2009.
34. A. Platzer and E.M. Clarke. The Image Computation Problem in Hybrid Systems Model Checking. In *HSCC*, pages 473–486, 2007.
35. A. Platzer and E.M. Clarke. Formal Verification of Curved Flight Collision Avoidance Maneuvers: A Case Study. In *FM*, pages 547–562, 2009.
36. A.M. Rabinovich. On the Decidability of Continuous Time Specification Formalisms. *J. Log. Comput.*, 8(5):669–678, 1998.
37. S. Sankaranarayanan and A. Tiwari. Relational abstractions for continuous and hybrid systems. In *CAV*, pages 686–702, 2011.
38. M. Sheeran, S. Singh, and G. Stålmarck. Checking Safety Properties Using Induction and a SAT-Solver. In *FMCAD*, pages 108–125, 2000.
39. A. Tiwari. Abstractions for hybrid systems. *Formal Methods in System Design*, 32(1):57–83, 2008.
40. S. Tonetta. Abstract Model Checking without Computing the Abstraction. In *FM*, pages 89–105, 2009.
41. Y. Yushtein, M. Bozzano, A. Cimatti, J.-P. Katoen, V.Y. Nguyen, Th. Noll, X. Olive, and M. Roveri. System-software co-engineering: Dependability and safety perspective. In *SMC-IT*, pages 18–25. IEEE CS Press, 2011.