# Taylor Model Flowpipe Construction for Non-linear Hybrid Systems

Xin Chen
RWTH Aachen University, Germany
xin.chen@cs.rwth-aachen.de

Erika Ábrahám
RWTH Aachen University, Germany
abraham@cs.rwth-aachen.de

Sriram Sankaranarayanan
University of Colorado, Boulder, CO
srirams@colorado.edu

***Abstract*—We propose an approach for verifying non-linear hybrid systems using higher-order *Taylor models* that are a combination of bounded degree polynomials over the initial conditions and time, bloated by an interval. Taylor models are an effective means for computing rigorous bounds on the complex time trajectories of non-linear differential equations. As a result, Taylor models have been successfully used to verify properties of non-linear continuous systems. However, the handling of discrete (controller) transitions remains a challenging problem.**

**In this paper, we provide techniques for handling the effect of discrete transitions on Taylor model flowpipe construction. We explore various solutions based on two ideas: *domain contraction* and *range over-approximation*. Instead of explicitly computing the intersection of a Taylor model with a guard set, *domain contraction* makes the domain of a Taylor model smaller by cutting away parts for which the intersection is empty. It is complemented by *range over-approximation* that translates Taylor models into commonly used representations such as template polyhedra or zonotopes, on which intersections with guard sets have been previously studied. We provide an implementation of the techniques described in the paper and evaluate the various design choices over a set of challenging benchmarks.**

## I. Introduction

The safety verification of software-enabled real-time control systems requires simultaneous reasoning about the control software as well as the physical environment into which it is embedded. *Hybrid systems* exhibit both discrete and continuous behavior. Therefore, they are a natural modeling formalism for such systems Although the reachability problem for hybrid systems is undecidable for all but the simplest subclasses [1], [2], in the recent years there has been much progress in the verification of affine hybrid systems, leading to tools such as SpaceEx [3]. However, very few practical verification approaches exist for the more general class of *non-linear* hybrid systems. In this paper, we present a promising approach using Taylor models.

*Taylor models (TM)*, originally developed by Berz and Makino [4], [5], [6], [7], approximate continuous and $k+1$ times differentiable functions by their Taylor polynomials of degree up to $k$ bloated by an interval representing the remainder. *TM arithmetic* lifts the basic arithmetic operations of addition, multiplication, division, time derivatives and anti-derivatives (integrals) over the underlying functions to their TM approximations. We use TMs to represent *flowpipes*, i.e., a set of states reachable by continuous dynamics from an initial set within a given time interval. As a result, they can be used to provide guaranteed enclosures to the solutions of ordinary differential equations (ODEs), often involving trigonometric and exponential functions. They have been observed to scale quite well and produce accurate results for complex systems [8].

The goal of this paper is to explore the use of TMs in the reachability analysis of hybrid systems using flowpipe construction [9], [10]. While TMs are well suited for handling ODEs in the flowpipe construction for purely continuous systems, their use in hybrid systems verification requires techniques for computing intersections of TMs with the guards of discrete transitions.

In this paper we propose techniques for handling the *intersections* of TMs with guard sets of discrete transitions. This operation forms a key primitive for the overall verification procedure for hybrid systems involving Taylor models to represent flowpipes. Our approach to compute intersections is based on two main ideas, which form the core contributions of this work:

1) We use *domain contraction*, inspired by interval constraint propagation (ICP) [11], to side-step the need to compute intersections with guards explicitly. Instead, domain contraction prunes away parts of the initial states and time intervals which lead to an empty intersection with the transition guard.
2) Since domain contraction is not always sufficiently accurate, we introduce *range over-approximation* methods which can be combined with domain contraction. We translate TMs into other widely used representations such as support functions [12], [3], template polyhedra [13] and zonotopes [14], [15], [12]. The intersections of these representations with the guard set has been well-studied by earlier approaches focussing mostly on affine hybrid systems. We translate the result of the intersection computation back to a TM for further flowpipe computations.

We have implemented the various techniques in this paper as a safety verification tool for hybrid systems. We evaluate the proposed approaches over a set of benchmark systems that are mostly beyond the reach of other currently available tools. One of the benchmarks used in our evaluation is a case-study for verifying closed-loop control systems proposed for con-

This work was funded, in part, by National Science Foundation (NSF) grants under award numbers CPS-1035845 and CNS-0953941. All views expressed are those of the authors and not necessarily of the NSF.

1052-8725/12 $26.00 © 2012 IEEE
DOI 10.1109/RTSS.2012.70

trolling plasma blood glucose levels in diabetic patients [16], [17]. The results show that our Taylor model approach is a promising direction for the verification of non-linear hybrid systems. A detailed description of the benchmarks used, the implementation and proofs of our results are available online [1].

**Related Work:** The safety verification of non-linear hybrid systems has been addressed using a wide variety of techniques in the past [2]. We discuss in the following, only the most closely related works.

Interval analysis techniques [18], [7], [19], [20] have been used to find validated solutions to ODEs. However, there have been fewer applications of these methods to hybrid systems, chiefly due to the problem of handling discrete transitions.

Henzinger et al. [21] used validated integration of ODEs inside modes and a simple handling of discrete transitions. Therein, a rectangular over-estimation of a flowpipe segment is used to detect if an intersection with a guard is possible. If so, the intersection's rectangular hull is transformed according to the transition and used as the starting set for the subsequent mode. However, such an approach is prone to gross over-approximations due to the wrapping effect. Similar ideas were proposed for rigorous event detection by Nedialkov et al. [22].

Recently, Ramdani et al. [23] proposed a technique that uses Taylor model interval methods to construct flowpipes for hybrid systems. Their work encodes the problem of detecting flowpipe guard intersections as a constraint-satisfaction problem, and proceeds to use interval constraint propagation. However, the key difference lies in the use of branch-and-prune to characterize the intersection accurately. In this paper, we use ICP to narrow down the possible ranges of initial conditions and the intersection times. However, instead of using branch-and-prune, we use range over-approximations.

Frameworks such as Newton, RealPaver, HySAT/iSAT, and CalCs implement some of the primitives needed for effective domain contraction [24], [25], [26], [27]. The key primitive of ICP has been used by the HySAT/iSAT solvers [24] to solve the reachability verification problem in terms of Boolean, differential and algebraic constraints. A key difference here lies in the use of guaranteed integration to construct flowpipes within modes: we use interval constraint propagation only when intersections with guards are computed but not in the flowpipe construction. Another difference is that our techniques apply range over-approximation instead of branch-and-prune to improve the result of interval constraint propagation.

Althoff et al. [28] propose an approach for avoiding intersction computations in flowpipe construction for linear hybrid systems that is similar to domain contraction. Their approach seeks computes a time to possibly reach the guard as a function of the initial state. The domain contraction technique proposed here differs in that it applies to non-linear systems and seeks to prune away initial states as well as time intervals.

Ratschan and She presented an interval-based technique for verifying hybrid systems in [29]. Here the state-space is discretized into finitely many rectangular cells and ICP is used to refine this abstraction to account for intersections with the transition guards and unsafe sets, thus avoiding unnecessary subdivisions. In contrast, we avoid subdividing the state-space in this work. Such a subdivision is often exponential in the number of state variables of the system.

TMs are also used in the Ariadne tool, which provides a rigorous framework for building hybrid systems verification tools based on the approximation of smooth functions [30]. Ariadne's core algorithm subdivides the state space into finitely many rectangular cells, where the degree of subdivision is determined dynamically according to some precision bounds. At each step, the flowpipes starting from previously reached cells are computed and those cells which intersect with these flowpipes are marked as reached.

Prabhakar et al. [31] present a scheme for reachability analysis for linear ODEs using degree-bounded polynomials to approximate the flow. The algorithm uses Bernstein polynomials (instead of Taylor polynomials used here) to approximate the time trajectories inside each mode. The approach therein is to sample some trajectories of the system and use Bernstein polynomials to compute piecewise approximations of the trajectories, keeping the overall error bounds to within $\epsilon$. However, the approach does not explicitly tackle the problem of intersections with guards, which is the main focus of our work. This is a significant consideration for tackling hybrid systems. The key idea of a priori bounding the overall error of the flowpipe to guide the degree of the approximation is quite attractive. However, such schemes are generally intractable in the presence of discrete transitions.

## II. Taylor Models

In this section, we present some preliminary notions of Taylor models. Further details are available from the works of Berz and Makino [4], [7], [5].

Let $\mathbb{N}$ be the set of natural numbers and $\mathbb{R}$ be the set of real numbers. We use $\mathbb{I}$ to denote the set of all intervals $I = [\ell, u] \subseteq \mathbb{R}$ with $\ell, u \in \mathbb{R}$ and $\ell \leq u$. Multi-dimensional intervals are Cartesian products of such single intervals. In the paper we also call them intervals.

A given polynomial $p$ is a *$k$-order (Taylor) approximation* of a function $f : D \to \mathbb{R}$ iff

(a) all partial derivatives of $f$ up to order $k$ exist and are continuous, denoted by $f \in C^k$, and
(b) $f(\vec{c}) = p(\vec{c})$ for the center point $\vec{c}$ of $D$ and for each $0 < m \leq k$, all of the order $m$ partial derivatives of $f$ and $p$ coincide at $\vec{c}$.

For any $f, g \in C^k$ and $k \geq 0$, we write $f \equiv_k g$ iff there is a polynomial $p$ which is a $k$-order approximation of both $f$ and $g$. Taylor models are based on the equivalence relation $\equiv_k$.

*Definition 2.1 (Taylor Model):* A *Taylor model (TM)* of order $k{>}0$ over a domain $D \in \mathbb{I}^n$ is a pair $(p, I)$ of a polynomial $p$ of degree at most $k$ over $n$ variables $\vec{x}$ and a *remainder interval* $I \in \mathbb{I}$. We say that $(p, I)$ is a *$k$-order over-approximation* of a function $f : D \to \mathbb{R}$, written $f \in (p, I)$, iff (i) $p \equiv_k f$ and (ii) $\forall \vec{x} \in D.\, f(\vec{x}) \in p(\vec{x}) + I := \{p(\vec{x}) + \vec{y} \,|\, \vec{y} \in I\}$.

[1] http://systems.cs.colorado.edu/research/cyberphysical/taylormodels/

Polynomials can be naturally extended to vectors to approximate functions $f : \mathbb{R}^n \to \mathbb{R}^m$. In what follows, we allow polynomials and TMs to have multi-dimensional ranges.

*Example 2.1:* Consider the transcendental function

$$f(x_1, x_2) = x_1 e^{-x_2} + 2x_2 \sin(x_1) \text{ with } (x_1, x_2) \in [-1, 1]^2 .$$

The TM $(p_3, I_3) = (x_1 + x_1x_2 + \frac{1}{2}x_1x_2^2, [-1, 1])$ is a 3-order over-approximation of $f$, and

$$(p_4, I_4) = \left(x_1 + x_1x_2 + \frac{1}{2}x_1x_2^2 - \frac{1}{6}x_1x_2^3 - \frac{1}{3}x_1^3x_2, [-0.1, 0.1]\right)$$

is a 4-order over-approximation. Generally, a higher order approximation yields a smaller remainder interval. ◁

A TM for a given function $f$ over a domain $D$ can be obtained by computing a degree $k$ Taylor polynomial $p_k$ for $f$ around $\vec{x} = \vec{c}$ and a conservative interval $I$ for the error $|f(\vec{x}) - p_k(\vec{x})|$ over $D$.

**Taylor Model Arithmetic:** TM arithmetic supports basic arithmetic operations of addition, scaling and multiplication of TMs over the same domain. Let $(p_1, I_1)$ and $(p_2, I_2)$ be two TMs. We define interval multiplication and addition as

$$[\ell_1, u_1] \otimes [\ell_2, u_2] := [\min_{c_i \in \{\ell_i, u_i\}} (c_1c_2), \max_{c_i \in \{\ell_i, u_i\}} (c_1c_2)] ,$$
$$[\ell_1, u_1] \oplus [\ell_2, u_2] := [\ell_1 + \ell_2, u_1 + u_2] .$$

For any given scalars $a, b$ we define

$$a(p_1, I_1) + b(p_2, I_2) := (ap_1 + bp_2, (a \otimes I_1) \oplus (b \otimes I_2))$$
$$(p_1, I_1) \cdot (p_2, I_2) :=$$
$$(p_1 \cdot p_2, (\mathsf{B}(p_2) \otimes I_1) \oplus (\mathsf{B}(p_1) \otimes I_2) \oplus (I_1 \otimes I_2)) ,$$

wherein $\mathsf{B}(p)$ denotes a safe interval enclosure of the range $p(D) := \{p(\vec{x}) \mid \vec{x} \in D\}$ of $p$ over $D$. Such an enclosure can be computed using standard interval arithmetic techniques [18]. Likewise, TM approximations of division $(p_1, I_1) \oslash (p_2, I_2)$ can be obtained, provided that $0 \notin \mathsf{B}(p_2) \oplus I_2$.

Note that for any $f_1 \in (p_1, I_1)$ and $f_2 \in (p_2, I_2)$ we have that $af_1 + bf_2 \in a(p_1, I_1) + b(p_2, I_2)$

Functions such as $\sqrt{x}, \sin(x), \cos(x), e^x$ and $\log(x)$ can be approximated by means of power series expansions. In turn, these expansions along with the models for elementary arithmetic operations provide a systematic approach for approximating any function specified symbolically as a TM.

A common convention used for TMs is to *normalize* the domain of a TM to $[-1, 1]^n$ using basic TM arithmetic, without changing the range defined.

For TMs $T_1, T_2$, we simply write $T_1 + T_2, T_1 \cdot T_2$ for the TMs defined by the TM arithmetic. For a TM $(p, I)$ over variables $(\vec{x}, \vec{y})$ and a TM $T$ we write $p(T, \vec{y}) + I$ for the TM resulting from $(p, I)$ by substituting $T$ in the place of $\vec{x}$.

**Order Lowering:** It is often necessary to conservatively simplify a given order $n$ TM $(p_n, I_n)$ to a lower order $k$ TM $(p_k, I_k)$, where $k < n$. To do so, we write $p_n = p_k + r_{n-k}$ where $p_k$ represents all monomials of degree $k$ or less in $p_n$ and $r_{n-k}$ represents the remaining terms. The simplified TM is given by $(p_k, I_n \oplus \mathsf{B}(r_{n-k}))$. Specifically, lowering the order to $n = 1$ results in the *linearization* of a TM.

However, the true utility of TMs lie in their ability to represent functions that are, in general, not known in a closed form. We will now discuss how TMs can be used to conservatively approximate the time trajectories of ordinary differential equations (ODEs) and hybrid systems.

## III. Taylor Model Integration

We will now outline the methodology for guaranteed integration of ODEs using TMs. The outline here is intended to serve as a tutorial introduction to TMs. Further details are discussed in, e.g., [5], [7], [4].

*Definition 3.1 (Initial Value Problem):* An instance of the *initial value problem* (IVP) consists of an ODE $\frac{d\vec{x}}{dt} = F(\vec{x}, t)$ over the state $\vec{x} \in \mathbb{R}^n$, an initial interval $\vec{x}(0) \in X_0 \in \mathbb{I}^n$ for the values of $\vec{x}$ at time 0 and a time interval $[t_1, t_2] \in \mathbb{I}$. The goal is to compute an enclosure $X_{[t_1,t_2]}$ of the states reachable from $X_0$ over the time interval $[t_1, t_2]$ according to the ODE.

We assume that the ODE $F(\vec{x}, t)$ is locally Lipschitz continuous, assuring existence and uniqueness of the time trajectories over some time interval $[-\Delta, \Delta]$ with $\Delta > 0$ and $[t_1, t_2] \subseteq [0, \Delta]$.

*Example 3.1:* Consider the IVP given by $\frac{dx}{dt} = 1 + x^2$ with initial interval $x(0) \in [0, \frac{1}{2}]$ and time interval $t \in [0, \frac{1}{10}]$. Throughout this section, we will demonstrate the application of TM integration over this running example.

Our approach to solving the IVP is to perform a flowpipe construction [9], [10]:

1) Starting with the initial set $X_0$, we compute an approximation $X_{[0,\delta_1]}$ for the states reachable from $X_0$ in time $[0, \delta_1]$. The set $X_{[0,\delta_1]}$ is called the first *flowpipe segment*.
2) Starting from the $i^{th}$ flowpipe segment $X_{[\delta_{i-1},\delta_i]}$, we advance our approximation to a flowpipe segment $X_{[\delta_i,\delta_{i+1}]}$ for some time $\delta_{i+1} > \delta_i$.
3) The answer to the IVP is given as the set union of the corresponding flowpipe segments that cover the interval $[t_1, t_2]$ of interest.

Flowpipe construction may be *fixed* timestep approximations wherein $\delta_{i+1} - \delta_i = h$, for an input $h > 0$, or *adaptive* wherein the time advance in each step is determined by considerations that may include proximity to the guards or the nature of the derivatives in a neighborhood.

**Flow Map:** The assumptions made for our IVPs guarantee the existence of a *flow map* $\vec{x}(t) = \varphi(\vec{x}_0, t)$ over $\vec{x}_0 \in X_0$ and some time range $t \in [-\Delta, \Delta],\ \Delta > 0$ that maps a given initial condition $\vec{x}(0) = \vec{x}_0$ to the state reached at time $t$ [32]. Nevertheless, the map $\varphi$ cannot be expressed in a known closed form for all but the simplest of ODEs. However, as we will show in the ensuing discussion, *it is possible to compute a TM $(p_k, I)$ of any given order $k \in \mathbb{N}$ for the map $\varphi$ for $\vec{x}_0 \in X_0$ and $t \in [-\Delta, \Delta]$.*

*Example 3.2:* Consider the ODE from Example 3.1. For this simple case, we have a closed form solution yielding the flow map $\varphi(x_0, t) = \tan(t + \tan^{-1}(x_0))$. Knowing this flow map, the Taylor series expansion around $t = 0$ yields an order 4 Taylor polynomial $p_4(x_0, t) = t + x_0 + x_0^2 t + x_0 t^2 + \frac{t^3}{3}$.

The TM for $\varphi$ is $(p_4, I)$ where $I$ is some safe enclosure of the error $\varphi(x_0, t) - p_4(x_0, t)$ over $x_0 \in [0, \frac{1}{2}]$ and $t \in [0, \frac{1}{10}]$.

The challenge here is to compute a conservative model $(p_k, I)$ for a flow map $\varphi$, *without knowing the flow map in a simple closed form*. This is key since most non-linear systems cannot be integrated (unlike our simple running example).

Assume that $X_0$ is given by a TM over some domain $D_0 \in \mathbb{I}^n$. For $i \geq 1$, the construction of the $i^{th}$ TM flowpipe proceeds in the following steps: **(1)** Determine a $k$-order approximation $p_k(\vec{x}_0, t)$ for the flow map $\varphi$ starting from the initial set $X_{i-1}$ (a TM) within the time interval $[0, \delta_i - \delta_{i-1}]$. The domain of $p_k$ is $\vec{x}_0 \in X_{i-1}$ and $t \in [0, \delta_i - \delta_{i-1}]$. **(2)** Evaluate a proper interval $I$ such that $(p_k, I)$ is an over-approximation of the $\varphi$. **(3)** The $i^{th}$ flowpipe $X_{[\delta_{i-1}, \delta_i]}$ is computed as the TM $p_k(X_{i-1}, t) + I$ by TM arithmetic. Then the initial set $X_i$ for the next flowpipe is the TM $p_k(X_{i-1}, \delta_i - \delta_{i-1}) + I$. TM order lowering is often necessary to keep the degrees of successive flowpipe segments low.

Therefore all of the flowpipes can be computed as the TMs over the domain $D_0$ and a time interval w.r.t. the corresponding time step.

**Computing the Polynomial $p_k$:** The polynomial $p_k$ can be computed in one of two ways: (a) applying Picard iteration over TMs for finitely many times, or (b) using a truncated Lie series by computing Lie derivatives of order $m$, wherein $0 \leq m \leq k$. Given an ODE $\frac{d\vec{x}}{dt} = F(\vec{x}, t)$ and a function $g(\vec{x}_0, t)$ the Picard operator defined by $F$ is given by

$$\mathbb{P}_F(g)(\vec{x}_0, t) = \vec{x}_0 + \int_0^t F(g(\vec{x}_0, s), s)\ ds\,.$$

It is well known that if $F$ is Lipschitz continuous then the operator $\mathbb{P}_F$ is contractive and its unique fixed point defines the solution to the ODE.

The Picard operator can be applied to derive a Taylor expansion $p_n$ of order $n$ for its fixed point. Let $p_F$ be the Taylor polynomial for function $F$. The TM Picard operator is given by $\mathbb{P}_F(g_i) = \vec{x}_0 + \int_0^t p_F(g_i(\vec{x}_0, s), s)\ ds$. The monomials of degree greater than $n$ in the result are dropped and the process is iterated until it converges. Convergence of this process follows, since each non-convergent iteration can be shown to result in the addition of at least one monomial whose degree is higher than the monomials encountered in the previous iteration. Therefore, at most $n$ Picard iterations are needed to generate an order $n$ Taylor expansion.

*Example 3.3:* Consider the ODE $\frac{dx}{dt} = 1 + x^2$ from Example 3.1. Table I shows the Picard iterates obtained for computing $p_4$, a 4-order approximation. At each iterative step, we retain the monomial terms of degree up to 4 in the result. Since $g_2 = g_3$, the iteration converges after 3 iterations.

An alternative approach is to use the Lie derivatives of $\vec{x}(t)$ to compute the Taylor expansion for the flow map. Recall that the Lie derivative of a function $g(\vec{x}, t)$ w.r.t a vector field $F(\vec{x}, t)$ is given by $\mathcal{L}_F(g) = (\nabla_x\ g) \cdot F(\vec{x}, t) + \partial_t g$. The Lie derivative gives the value of the time derivative of the function $g(\vec{x}(t), t)$ applied to any time trajectory $\vec{x}(t)$. We may

TABLE I
THE COMPUTATION OF 4-ORDER APPROXIMATION TO FLOW MAP BY PICARD ITERATION.

| Picard Iterate | Order 4 Taylor polynomial |
|---|---|
| $x_0$ | $g_0:\ x_0$ |
| $x_0 + \int_0^t (1 + g_0^2) dt$ | $g_1:\ x_0 + t + x_0^2 t$ |
| $x_0 + \int_0^t (1 + g_1^2) dt$ | $g_2:\ x_0 + t + x_0^2 t + x_0 t^2 + \frac{t^3}{3}$ |
| $x_0 + \int_0^t (1 + g_2^2) dt$ | $g_3:\ x_0 + t + x_0^2 t + x_0 t^2 + \frac{t^3}{3}$ |

therefore write the truncated series as

$$g_n(\vec{x}_0, t) = g(\vec{x}_0, 0) + \mathcal{L}_F(g(\vec{x}_0, 0))t + \mathcal{L}_F^2(g(\vec{x}_0, 0))\frac{t^2}{2!} + \cdots + \mathcal{L}_F^n(g(\vec{x}_0, 0))\frac{t^n}{n!}\,.$$

Our goal is to derive a Taylor expansion for $\vec{x}(t)$ as a function of $\vec{x}_0$ and $t$. To this end, we apply the truncated Lie series for each function $f_j(\vec{x}) = x_j$, the $j^{th}$ component of $\vec{x}$.

*Example 3.4:* Consider the ODE $\frac{dx}{dt} = 1 + x^2$ given in Example 3.1. We wish to compute the polynomial $p_4$ representing the Taylor expansion of degree 4 for the flow map which is assumed not to be known. The Lie derivatives are:

$$\mathcal{L}(x):\ 1 + x^2,\ \ \mathcal{L}^2(x): 2x + 2x^3,\ \ \mathcal{L}^3(x):\ 2 + 8x^2 + 6x^4,$$
$$\mathcal{L}^4(x):\ 16x + 40x^3 + 24x^5\,.$$

To construct $p_4$, we consider the expansion around $x = x_0$ and $t = 0$, i.e.,

$$g_4(x_0, t) = x_0 + (1 + x_0^2)t + (2x_0 + 2x_0^3)\frac{t^2}{2!} + (2 + 8x_0^2 + 6x_0^4)\frac{t^3}{3!} + (16x_0 + 40x_0^3 + 24x_0^5)\frac{t^4}{4!}\,.$$

Next, we compute a polynomial of order 4 for the truncated Lie series. Since the RHS of the ODE is a polynomial in this case, this operation simply consists of dropping monomials of degree greater than 4. As a result, we obtain, $p_4(x_0, t) = x_0 + t + x_0^2 t + x_0 t^2 + \frac{t^3}{3}$. Not surprisingly, this is the same result as the Picard iteration.

**Computing the Remainder Interval:** Thus far, we have computed a Taylor expansion $p_k$ of order $k$ for the flow map $\varphi$. Our goal is to derive a bound for the error $e(\vec{x}_0, t) = \varphi(\vec{x}_0, t) - p_n(\vec{x}_0, t)\ \ \vec{x}_0 \in X_0, t \in [0, \Delta]$.

A standard approach is to find an interval $I$ such that evaluating the Picard operator over $(p_k, I)$ yields a TM of the form $(p_k, J)$ wherein $J \subseteq I$. In other words, we wish to find an interval $I$ over which the Picard iteration is *contractive*. This proceeds by first choosing an initial interval $I_0$ (an estimate) and computing the Picard operator over $p_k + I_0$ to yield $(p_k', J)$. We then compute an enclosure of $p_k' - p_k + J$. If the resulting enclosure is contained in $I_0$, then we stop and declare $I_0$ to be a valid over-approximation. Otherwise, we simply expand $I_0$ by multiplying it with a suitable scale factor. If the Picard operator is contractive on a computed enclosure $I_n$, we can further refine this by repeatedly applying Picard iteration to it [2] [7].

[2]This process is reminiscent of widening and narrowing in abstract interpretation to accelerate convergence to a fixed point [33].

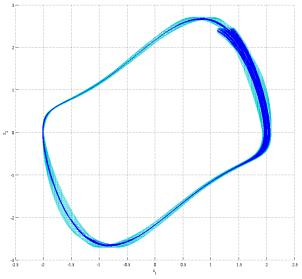


Fig. 1. Taylor model flowpipes for the Van-der-Pol Oscillator.

*Example 3.5:* Consider the running example again. We wish to consider the Taylor model $(p_4, [-1, 1])$. We first carry out the Picard iteration given by $p_4' = x_0 + \int_0^t (1 + (p_4 + I_0)^2) dt$. We obtain $p_4' = p_4 + \int_0^t ([0, 0.107] \oplus [-2.96, 2.96] \oplus [0, 1]) dt = p_4 + [-2.96, 4.067] \otimes [0, \frac{1}{5}]$. This yields, $p_4' = p_4 + [-0.6, 0.82]$. As a result, the Picard operator is contractive on $(p_4, [-1, 1])$. This suggests that $(p_4, [-1, 1])$ is a valid over-approximation over the time interval $[0, 0.2]$. In fact, using the Picard iteration suggests that $(p_4, [-0.6, 0.82])$ is also a valid over-approximation. The Picard operator is repeatedly applied until no significant improvement of the error interval is seen.

The remainder interval is a potential source of inaccuracies due to the *wrapping effect* [18]. Several techniques have been proposed to overcome this limitation [34], [35], [20].

*Example 3.6 (Van-der-Pol Oscillator):* The Van-der-Pol Oscillator is defined by $\frac{dx}{dt} = y$, $\frac{dy}{dt} = y - x - x^2 y$ with the initial set $x(0) \in [1.1, 1.4]$ and $y(0) \in [2.35, 2.45]$. Figure 1 shows the TM flowpipe constructed until oscillation is detected. The simulation trajectories (dark blue) are overlaid on the TM approximation.

### A. Hybrid Systems Verification

We now briefly recall the overall strategy for exploring behaviors of a hybrid system using the presented TM flowpipe construction and motivate the problem of handling discrete transitions. Hybrid systems are formalized by hybrid automata [36], [1], [2]. Flowpipe construction itself is a widely-studied approach to hybrid systems verification. We refer the reader to earlier works and references therein, for an exposition of flowpipe construction techniques [10], [3].

A hybrid automaton consists of a finite set of modes $Q$ and continuous variables $\vec{x}$ that belong to a state space $X \subseteq \mathbb{R}^n$. The dynamics of the system are defined by means of ODEs specifying the dynamics of $\vec{x}$ in each mode and by discrete transitions between modes modeling instantaneous mode changes. A state $\langle q, \vec{v} \rangle$ of a hybrid automaton represents a mode $q$ and a valuation $\vec{v}$ to the continuous variables $\vec{x}$.

*Definition 3.2 (Safety Verification):* Given a hybrid system $\mathcal{H}$ and a set of unsafe states $U$, the *safety verification* problem is to decide if any state in $U$ is reachable from some initial state of $\mathcal{H}$.

A popular approach to safety verification is through *flowpipe construction*, wherein the reachable states of the hybrid system are explored symbolically and incrementally. Then we check the emptiness of the intersection between the flowpipes and the unsafe set. Flowpipe construction for hybrid systems consists of two key primitives: (1) *Time elapse:* Given an initial set of states $S_0$ all belonging to mode $q$, compute the states reachable inside mode $q$ due to the continuous evolution of state variables $\vec{x}$ according to the ODE $\frac{d\vec{x}}{dt} = F_q(\vec{x}, t)$.

(2) *Image computation:* Given a set of states $S$ inside mode $q$ and a transition $\tau : \langle q, q', G, P \rangle$, check if $\tau$ is enabled on any state in $S$ and if so, compute the image $S'$ of $S$ w.r.t $\tau$.

### B. Image Computation

Image computation across a discrete transition has two parts: (a) compute the intersection of the segment $S_j$ with the guard set $G$ of the transition; and (b) compute the image of the set $S_j \cap G$ across the transformation $\vec{x}' = P(\vec{x})$.

**Intersection Computation:** Given a TM $(p_j, I_j)$ for segment $S_j$ and a guard set $G$, we wish to consider techniques for computing the intersection $S_j \cap G$. A straightforward approach is to conjoin the guard constraint for $G$ to the TM $(p_j, I_j)$.

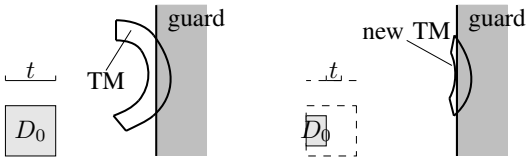
However, manipulating such *constrained* TMs inside a flowpipe construction scheme can be quite cumbersome: (a) it is unclear how arithmetic on such models can be performed, or the guaranteed integration scheme can be adapted; and (b) simply carrying around constraints does not solve the problem of having to check emptiness and subsumption during the verification process.

In this paper, we present two classes of strategies:

**Domain Contraction:** The constraint $\vec{x} \in G$ can be used to provide a *contraction* of the domain $D_0$ and time interval for $t$. This results in a conservative TM approximation of the set $S_j \cap G$. Domain contraction uses ideas from interval constraint propagation (ICP), and can be applied to guard sets $G$ specified by non-linear constraints.

**Range Over-Approximation:** We transform the $S_j$ into a different representation $P_j$ such as a zonotope or support function, and use standard techniques over these representations to over-approximate the intersection $P_j \cap G$. Next, we transform the intersection back into a TM. Each step is carried out conservatively, so that the overall result is an over-approximation.

**Image across Transformation:** Let $(p, I)$ represent the TM for $S_j \cap G$. We wish to compute the image across the reset map $\vec{x}' = P(\vec{x})$. This is achieved first computing the Taylor polynomial $r_k$ for $P$ up to some order $k$. The Taylor polynomial for the result is given by the composition $r_k(p(\vec{x}_0, t))$. Next, we compute the interval error by over-approximating the error $e = P(p(\vec{x}_0, t) + I) - r_k(p(\vec{x}_0, t))$ over the intervals bounding $\vec{x}_0$ and $t$. We can use standard interval arithmetic techniques to estimate safe interval bounds $I_e$ on the error $e$ [18]. The overall TM for the image is given by $(r_k(p), I_e)$.




(a) The original TM (b) After the domain contraction

Fig. 2. Illustration of the domain contraction method.

## IV. Intersections of Taylor Models and Guards

We now present techniques for domain contraction and range over-approximation.

### A. Domain Contraction

Domain contraction *side-steps* the need to compute an intersection of the TM segment with the guard by computing an approximation of the time intervals and starting states inside the mode which lead to an intersection with the guard for a particular enabled discrete transition.

Consider a TM flowpipe $S_j : (p_j(\vec{x}_0, t), I_j)$ over $\vec{x}_0 \in D_0,\ t \in J$, and a guard set $G$ defined by predicate $\gamma(\vec{x})$. The intersection is given by

$$\underbrace{\vec{x} = p_j(\vec{x}_0, t) + I_j \ \wedge\ \vec{x}_0 \in D_0 \ \wedge\ t \in J}_{\text{TM Flowpipe Segment}} \ \wedge\ \underbrace{\gamma(\vec{x})}_{\text{Guard}} .$$

The constraint $\gamma(\vec{x})$ on the range implies as smaller domain $D_0' \subseteq D_0$ for $\vec{x}_0$ and $J' \subseteq J$ for $t$. The contracted TM $(p_j(\vec{x}_0, t), I_j)$, where $\vec{x}_0 \in D_0',\ t \in J'$ is an over-approximation of $S_j \cap G$.

If the contraction procedure results in an empty domain, then we conclude that the intersection is empty. We illustrate the process of domain contraction by a simple example and discuss algorithms and heuristics for domain contraction.

*Definition 4.1 (Domain Contraction):* Let $(p_j, I_j)$ be a TM flowpipe, $\vec{x}_0 \in D_0, t \in J$, and $\gamma(\vec{x})$ be a predicate over $\vec{x}$. Subsets $D_0' \subseteq D_0, J' \subseteq J$ are valid contracted domains for $\vec{x}_0, t$ iff the following entailment holds:

$$\vec{x}_0 \in D_0 \wedge t \in J \wedge \vec{x} \in p_j(\vec{x}_0, t) + I \wedge \gamma(\vec{x}) \models \vec{x}_0 \in D_0' \wedge t \in J' .$$

*Example 4.1 (Domain Contraction):* We show an example of domain contraction by using ICP. Consider a TM

$$x_1 = -5x_0 + x_0 t + [-0.1, 0.1]$$
$$x_2 = x_0 t - t^2 + [-0.1, 0.1]$$

wherein $\vec{x}_0 \in [-1, 0.01]$ and $t \in [0, 0.8]$. The guard set is given by $\gamma(x_1, x_2) :\ x_1 \leq x_2$. Substituting $x_1, x_2$ in terms of $x_0, t$ on the guard predicate, we obtain

$$-5x_0 + [-0.1, 0.1] \leq -t^2 + [-0.1, 0.1]$$

This yields $t \in [0, 0.5]$. We propagate this new constraint to contract the range of $x_0$ and obtain $x_0 \in [-0.4, 0.01]$. The contracted domain is $t \in [0, 0.5] \ \wedge\ x_0 \in [-0.4, 0.01]$.

We now discuss algorithms for domain contraction. Domain contraction will be repeatedly performed for each flowpipe segment to determine if an intersection with a guard exists. It is therefore of great importance to consider algorithms that trade-off precision in terms of avoiding spurious intersections with the need to keep the running time as small as possible.

**Contraction as Optimization:** We first note that domain contraction is a non-linear optimization problem, where we wish to compute bounds on each component of $\vec{x}_0$ and $t$ subject to the non-linear constraints $\vec{x}_0 \in D_0, t \in J, \vec{x} \in p_j(\vec{x}_0) + I, \gamma(\vec{x})$, encoding the domain, the flowpipe segment and the guard, respectively.

It is well-known that general non-linear programming is NP-hard. Most approaches that find local optima using gradient descent cannot be used in this setting to yield a sound verification procedure. Techniques such as Sum-of-Squares (SOS) Relaxation [37] can be used to achieve domain contraction. However, our experience with using SOS programming for domain contraction problems encountered during flowpipe construction is that (a) SOS programming can be often quite slow, leading to an unacceptable slowdown in the overall flowpipe computation and (b) numerical errors in the solution may propagate through the flowpipe construction, ultimately leading to an unsound result.

**Interval Constraint Propagation:** Instead of using nonlinear solvers, we contract the interval for $\vec{x}_0, t$ using interval constraint propagation (ICP) [11]. Example 4.1 illustrates an application of this idea. We first express the constraints in terms of $\vec{x}_0$ and $t$ by substituting $p_j(\vec{x}_0) + \vec{u}$ for $\vec{x}$, where $\vec{u} \in I$ is a fresh set of variables to represent uncertainty. We can now use ICP to contract the domain $\vec{x}_0$ and $t$ until the overall set of constraints are hull consistent. Using ICP for domain contraction is easy to implement and has been found to be a very effective trade-off in our experiments.

**Contraction Using Branch-and-Prune:** The process of contraction using ICP can be improved by using branch-and-prune. This is implemented by partitioning the result of ICP into many subsets and performing contraction recursively on each subset. The resulting contraction is obtained as a disjoint union of many domains $D_1' \cup \ \ldots \cup D_m'$. There are various methods which can be used to compute such a non-convex contraction, such as the one from Ramdani et al. [23].

The complexity of using branch-and-prune can be exponential in the depth of the recursive calls. Another source of complexity lies in the creation of a large number of tiny flowpipe segments, all of which need to be propagated as part of the overall construction.

**An Efficient Trade-Off:** An efficient trade-off between the various approaches presented here is to perform interval contraction with branch-and-prune until the intervals are smaller than some cutoff width $\epsilon$. The main difference between our approach and the standard interval branch-and-prune approach [11] is that we only keep one subdivision in every branching step, as presented by Algorithm 2.

(a) We denote by $\vec{y}$ the composition $\langle \vec{x}_0, t \rangle$. The algorithm computes the bound for each component $(\vec{y})_j$ of $\vec{y}$ separately for $j = 1, \ldots, n+1$, combining the disjoint intervals

**Input:** TM $(p_j(\vec{y}), I_j)$ with domain $\vec{y} \in D$.
**Output:** The contracted domain.

1: **for all** $i = 1, \ldots, n$ **do**
2: Compute a conservative approximation *lo* for the lower bound of the $(D_{\min})_i$;
3: Compute a conservative approximation *up* for the upper bound of the $(D_{\min})_i$;
4: **if** $lo \leq up$ **then**
5: Update $(D)_i$ to $[lo, up]$.
6: **else**
7: Set $D$ empty and break;
8: **end if**
9: **end for**
10: **return** $D$;

**Algorithm 1:** Main procedure of domain contraction

**Input:** TM $(p_j(\vec{y}), I_j)$ with domain $\vec{y} \in D$, $i$ and $\epsilon$.
**Output:** Approximation of the lower bound in dimension $i$.

1: Set *lo* as the lower bound of $(D)_i$;
2: Set *up* as the upper bound of $(D)_i$;
3: **while** the size of $[lo, up]$ is larger than $\epsilon$ **do**
4: Split $[lo, up]$ into $[lo, a]$ and $[a, up]$ wherein $a = \frac{lo+up}{2}$;
5: **if** $(p_j(\vec{y}), I_j)$ with $\vec{y} \in D$ and $(\vec{y})_i \in [lo, a]$ intersects the guard **then**
6: $up \leftarrow a$;
7: **else**
8: $lo \leftarrow a$;
9: **end if**
10: **end while**
11: **return** $lo$;

**Algorithm 2:** Lower Bound approximation search.

that could potentially contain a solution into a single interval.

(b) Each step consists of a branch-and-prune over the dimension $j$ for which a lower bound is currently being searched.

(c) Finally, the contracted lower bound for $(\vec{y})_i$ is used in turn to improve future contractions for $(\vec{y})_j$ for $j > i$.

Another key improvement is the use of linear programming (LP) solvers in addition to ICP to improve efficiency. This is achieved by lowering the order of all TMs to degree 1, making them affine constraints. LP solvers are then used to successively compute bounds for each of the dimensions $(\vec{x}_0)_j$, where $1 \leq j \leq n$ and the time variable $t$. The procedures are summarized by the Algorithms 1 and 2.

*Theorem 4.1:* Algorithm 1 yields a valid domain contraction according to Definition 4.1.

*Example 4.2:* We consider once again the Van-der-Pol system from Example 3.6. This time, we add a guard set $x \geq 2$. Figure 3 shows the result of domain contraction on the flowpipes generated. It is interesting to see that the result of contraction eliminates unreachable states that are in the intersection of the original flowpipe segments and the guards. The result also includes states that are not in the guard set.

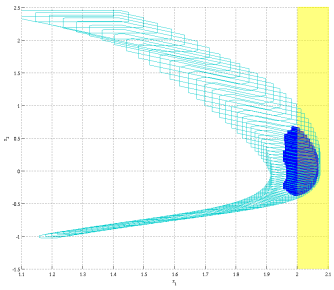


Fig. 3. Domain contraction for guard sets.

This will be improved by combining domain contraction with range over-approximations.
Finally, we note that the process of domain contraction can also be used to handle invariant sets that are commonly specified for hybrid automata modes.

### *B. Range Over-Approximation*

Range over-approximation seeks a temporary change of representations, using geometric objects like template polyhedra, zonotopes and support functions on which intersection computation with guard sets have been well-studied.

**Support Functions:** A support function over-approximation of a set $S$ provides for every input direction $\vec{l}$ an *intercept* $z_l$ such that $\vec{x} \in S \models \vec{l}^T\vec{x} \leq z_l$. Template polyhedra are support functions wherein the domain of directions $\vec{l}$ belong to a finite set of template directions $L$. Template polyhedra and support functions have been used to construct flowpipes for affine hybrid systems [13], [3], [12]. The transformation from TMs to support functions allows us to use the available methods [12] for intersections.

The construction of support function (or template polyhedral) over-approximation for a Taylor model flowpipe segment $S : p(\vec{x}_0, t) + I$ is conceptually simple. Once again, we may carry out the optimization $z_l : \max \vec{l}^T\vec{x}$ s.t. $\vec{x} \in p(\vec{x}_0, t) + I \wedge \gamma(\vec{x})$. If the degree of $p$ is high, this is a general nonlinear, non-convex mathematical programming problem. Also here, global optimization is intractable. However, numerous techniques can be used to obtain an upper bound for $z_l$, yielding a sound support function over-approximation. Given that the interval ranges for $\vec{x}_0, t$ are known, it is possible to evaluate $\vec{l}^T$ directly using interval arithmetic. However, this cannot take the guard $\gamma$ into account, unless $\gamma$ contains a constraint of the form $\vec{l}^T\vec{x} \leq c$. A better strategy for *polyhedral guard sets* is to formulate the optimization as a linear program by linearizing the TM.

**Zonotopes:** We may also use the methods proposed for zonotopes to handle TM/guard intersections. A zonotope can be viewed as the image of a hyper-rectangle under an affine mapping [38]. Therefore we have the following lemma showing the connection between zonotopes and TMs.

*Lemma 4.1:* A zonotope is a TM of order 1. Every TM lowered to order 1 yields a zonotope.

A zonotope with generator-based representation can be easily transformed to a TM. Given a zonotope $\mathcal{Z} = (c, \langle g_1, \ldots, g_m \rangle)$, the set of $\mathcal{Z}$ can be represented by the TM $(p(\vec{x}), [0,0]^m)$ such that $p(\vec{x}) = c + (g_1, \ldots, g_m)\vec{x}$ and $\vec{x} \in [-1,1]^n$. The other direction is also not difficult, given an order 1 TM $(p, I)$, we compute the generator-based representations for the set $p$ and $I$ respectively and then sum them up. The conversion from a high order TM to a zonotope proceeds by (a) performing a degree reduction procedure that computes a safe enclosure of the non-linear terms and (b) casting the resulting set as a zonotope. Hence, a TM/guard intersection can be handled by (1) over-approximating the TM by a zonotope, (2) computing a zonotopic enclosure for the zonotope/guard intersection by some available method, (3) transforming the enclosure back to a TM.

## V. Implementation and Evaluation

We now briefly discuss the implementation of the ideas in this paper and present an experimental evaluation over some benchmark problems.

**Implementation:** The techniques presented thus far have been implemented in C++ using a guaranteed interval arithmetic library supported by the exact arithmetic package MPFR. Our implementation currently supports non-linear hybrid systems with polynomial dynamics inside modes and discrete transitions with polyhedral guards. The implementation of guaranteed integration for handling dynamics inside a mode uses the basic algorithm of Berz and Makino [5] with numerous modifications for supporting its application to hybrid systems. Domain contraction is implemented using ICP. We have also implemented range over-approximations using support functions, template polyhedra and zonotopes. Finally, we implemented *preconditioned* TMs, wherein a linear change of basis transformation improves the precision and the efficiency of the computation [35].

**Evaluation:** We first describe the results on some of the benchmark instances and a comparison with the Ariadne tool [30], which is perhaps the only available tool that handles non-linear hybrid systems robustly. Table II shows the results on some benchmark examples taken from related approaches to verifying non-linear hybrid systems [39] and some control systems for glycemic control in diabetic patients [16], [17]. All experiments were tested on a i7-860 2.8GHz CPU with 4GB RAM running Ubuntu Linux.

**Glycemic Control in Diabetic Patients:** We formulated a simple hybrid model of glycemic control in diabetic patients following the description of feedback control strategies by Fisher [17] and Furler et al. [16]. The dynamics of insulin glucose in a type I diabetic patient is modeled by the Bergman minimal model with three state variables $(G, I, X)$ wherein $G$ is plasma glucose concentration above the basal value $G_B$ and $I$ is the plasma insulin concentration above the basal value $I_B$.

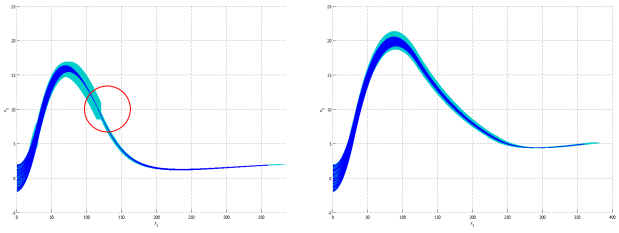


Fig. 4. Flowpipes constructed for the glycemic control schemes of (left) Furler et al. [16] and (right) Fisher [17].

$X$ is the insulin concentration in an interstitial chamber.

$$\begin{array}{rcl} \frac{dG}{dt} & = & -p_1 G - X(G + G_B) + g(t) \\ \frac{dX}{dt} & = & -p_2 X + p_3 I \\ \frac{dI}{dt} & = & -n(I + I_b) + \frac{1}{V_I} i(t)\,. \end{array}$$

Typical parameter values are $p_1 = 0.01, p_2 = 0.025, p_3 = 1.3 \times 10^{-5}, V_I = 12, n = 0.093$, $G_B = 4.5, I_b = 15$. The functions $g(t)$ and $i(t)$ model the infusion of glucose and insulin into the bloodstream. The insulin control scheme $i_1(t)$ due to Furler et al. [16] and $i_2(t)$ due to Fisher [17] assume that $G(t)$ is measured using frequent blood glucose tests (assumed to be error free):

$$i_1(t) = \begin{cases} \frac{25}{3} & G(t) \leq 4 \\ \frac{25}{3}(G(t) - 3) & G(t) \in [4, 8] \\ \frac{125}{3} & G(t) \geq 8 \end{cases}$$

$$i_2(t) = \begin{cases} 1 + \frac{2G(t)}{9} & G(t) < 6 \\ \frac{50}{3} & G(t) \geq 6 \end{cases}\,.$$

The influx of glucose after a meal is modeled as:

$$g(t) = \begin{cases} \frac{t}{60} & t \leq 30 \\ \frac{120 - t}{180} & t \in [30, 120] \\ 0 & t \geq 120 \end{cases}\,.$$

The initial conditions are $G(0) \in [-2, 2], X(0) = 0, I(0) \in [-0.1, 0.1]$. Our goal is to simulate the system for a total of $t = 360$ minutes to conclude upper and lower limits on the value of $G(t)$. Modeling of $g(t)$ and $i(t)$ values yields a 9 mode hybrid automaton for the control scheme $i_1(t)$ and 6 modes for the control scheme $i_2(t)$. Figure 4 shows the constructed TM flowpipes (plotted by their box enclosures) along some simulated trajectories (in blue color). In the red circle, we can see that the intersection of the guard and box enclosures is much larger than our intersection over-approximation.

**Vehicle Model:** The dynamics of a non-holonomic vehicle is given as follows,

$$\begin{array}{lll} \frac{dx}{dt} = vc_t & \frac{dy}{dt} = vs_t & \frac{dv}{dt} = u_1 \\ \frac{dc_t}{dt} = \sigma v^2 s_t & \frac{ds_t}{dt} = -\sigma v^2 c_t & \frac{d\sigma}{dt} = u_2 \end{array}$$

where $u_1, u_2$ are control inputs. We consider the case of a vehicle with three control modes $m_1, m_2, m_3$. The control inputs are given by $(u_1, u_2) = (-0.05, -0.1)$, for mode $m_1$, $(0, 0)$ for $m_2$ and $(0.05, 0.1)$ for $m_3$.The transitions between

TABLE II

EXPERIMENTAL RESULTS OVER SOME BENCHMARK SYSTEMS. LEGEND – DEG: DEGREE OF THE DYNAMICS, LOC: NUMBER OF LOCATIONS, VAR: NUMBER OF VARIABLES, $\delta$: TIME STEP, T: TIME HORIZON, ORD: ORDER OF THE TMS, T.T.: TOTAL TIME (S), T.I.: TIME OF COMPUTING INTERSECTIONS (S), MEM: MEMORY USED (MB), D.C.: DOMAIN CONTRACTION, R.M.: RANGE OVER-APPROXIMATION METHOD, Z: ZONOTOPES, S.F.: SUPPORT FUNCTIONS, EXC: THREW AN EXCEPTION, NR: CANNOT FIND A PROPER REMAINDER.

| | | | | Our tool | | | | | | | | Ariadne | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Benchmark | DEG | LOC | VAR | $\delta$ | T | ORD | T.T. | T.I. | MEM | D.C. | R.M. | T.T | MEM |
| Brusselator | 3 | 1 | 2 | 0.05 | [0,10] | 4 | 77 | 0 | 4 | - | - | 34 | 12 |
| Brusselator | 3 | 1 | 2 | 0.03 | [0,15] | 4 | 152 | 0 | 8 | - | - | EXC | - |
| Watertank | 1 | 4 | 2 | 0.1 | [0,80] | 3 | 9 | 5 | 8 | $\surd$ | Z | 7 | 24 |
| Van-der-PolE | 3 | 2 | 3 | 0.01 | [0,6] | 4 | 71 | 37 | 4 | $\surd$ | Z | EXC | - |
| Lotka-Volterra | 2 | 1 | 3 | 0.01 | [0,3] | 4 | 14 | 0 | 8 | - | - | 52 | 8 |
| Hallstah | 3 | 1 | 2 | 0.01 | [0,7] | 3 | 19 | 0 | $\leq$1 | - | - | 0.6 | $\leq$1 |
| B.B. no drag | 1 | 1 | 4 | 0.02 | [0,3] | 3 | 0.8 | 0.3 | $\leq$1 | $\surd$ | - | 0.2 | $\leq$1 |
| B.B. const drag | 1 | 2 | 4 | 0.02 | [0,3] | 3 | 2 | 0.8 | $\leq$1 | $\surd$ | - | 0.6 | $\leq$1 |
| B.B. Stokes-Einstein | 2 | 2 | 4 | 0.02 | [0,3] | 3 | 8 | 2 | $\leq$1 | $\surd$ | - | EXC | - |
| Diabetic [16] | 2 | 9 | 4 | 0.02 | [0,360] | 9 | 2138 | 663 | 430 | $\surd$ | Z | EXC | - |
| Diabetic [17] | 2 | 6 | 4 | 0.02 | [0,360] | 9 | 1804 | 443 | 410 | $\surd$ | Z | EXC | - |
| Watt governor [40] | 4 | 1 | 5 | 1e-4 | [0,15] | 5 | NR | - | - | - | - | EXC | - |
| Vehicle | 4 | 3 | 6 | 0.1 | [0,10] | 9 | 85 | 40 | 4 | $\surd$ | S.F. | EXC | - |
| Angiogenesis [39] | 2 | 1 | 12 | 1e-8 | [0,2e-6] | 4 | 34 | 0 | 12 | - | - | EXC | - |
| Coll-avoid-2 [41] | 2 | 3 | 12 | 0.01 | [0,10] | 3 | 27 | 8 | 3 | $\surd$ | - | EXC | - |

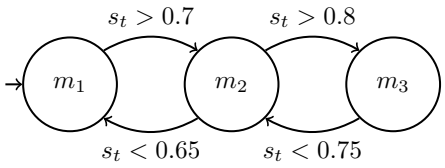



Fig. 5. Hybrid automaton of the vehicle model.

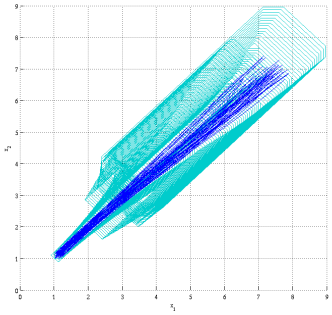


Fig. 6. Flowpipe constructed for the vehicle model.

modes are shown in Figure 5. We start the TM flowpipe construction from $m_1$ with the initial variable values

$$x \in [1, 1.2] \quad y \in [1, 1.2] \quad v \in [0.8, 0.81]$$
$$s_t \in [0.7, 0.71] \quad c_t \in [0.7, 0.71] \quad \sigma \in [0, 0.05]\,.$$

The projection of the result on $x, y$ is shown in Figure 6, where the TM flowpipes are plotted by their octagon over-approximations, and the simulation trajectories are in blue. This model is a good evaluation platform for the flowpipe/guard intersection techniques. Since lots of flowpipes intersect the guard in each mode, if the intersections can not be accurately over-approximated, the overestimation would increase drastically in the forthcoming integration steps.

**Angiogenesis Model:** We consider the Angiogenesis model which is presented in [42]. It is a non-linear ODE with 12 state variables. The initial conditions and parameters are as reported in [39]. The TM approach is able to construct a flowpipe with a time step as small as $10^{-8}$. The TM construction fails if a larger time step is used.

**Collision Avoidance:** We apply our method to the algebraic abstraction of the collision avoidance system analyzed recently by Platzer et al. [43] and earlier in [44] and [41]. We consider the instance of two aircrafts. The abstracted hybrid automaton has 3 modes and 12 variables. Initially, the distance between the two aircrafts is 20. We define the alert distance by 5 and the other parameters are same as those in [41]. We performed the TM flowpipe construction for the time horizon $[0, 10]$ and detected no collision. The results are given in Table II.

In addition, we consider many other benchmarks including the bouncing ball with various kinds of air friction, the Van-der-PolE and Hallstah examples from the HSolver suite [3], the Lotka-Volterra and the Brusellator systems.

**Comparison with Other Tools:** We compared our technique with the tools HySAT/iSAT [24] and Ariadne [30]. HySAT/iSAT both failed to complete on most of the benchmarks presented here (to be fair, HySAT/iSAT are meant to be generic non-linear constraint solvers). Table II shows the comparison with Ariadne. We note that our approach produced results in many cases where Ariadne fails due to too many subdivisions of the state-space. On the other hand, whenever Ariadne was able to complete, it computed the reach-set notably faster than our method. The implementations reported by Ramdani et al. [23] and Prabhakar et al. [31] were unavailable for comparisons at the time of writing.

## VI. CONCLUSION

In conclusion, we have presented techniques for hybrid system verification using Taylor models combined with domain

[3] http://hsolver.sourceforge.net/benchmarks

contraction and range over-approximations. Our approach has demonstrated a lot of promise on the benchmarks used in our evaluation. However, a lot remains to be done to make nonlinear hybrid system verification truly practical. For instance, the error in TM flowpipes can be considerable if the initial set is large. A standard approach there is to subdivide the initial set and perform the analysis separately on each subset. We are investigating *template TMs* to reuse work in this setting. We are also investigating integration of our tools inside environments such as Simulink/Stateflow, to enable direct evaluation on challenging control system benchmarks.

## REFERENCES


[1] J. Lygeros, "Lecture notes on hybrid systems," 2004, eNSIETA course.

[2] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer, 2009.

[3] 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 *Proc. CAV'11*, ser. LNCS, vol. 6806. Springer, 2011, pp. 379–395.

[4] M. Berz, *Modern Map Methods in Particle Beam Physics*, ser. Advances in Imaging and Electron Physics. Academic Press, 1999, vol. 108.

[5] K. Makino and M. Berz, "Taylor models and other validated functional inclusion methods," *J. Pure and Applied Mathematics*, vol. 4, no. 4, pp. 379–456, 2003.

[6] ——, "Rigorous integration of flows and ODEs using Taylor models," in *Proc. SNC'09*, 2009, pp. 79–84.

[7] M. Berz and K. Makino, "Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models," *Reliable Computing*, vol. 4, pp. 361–369, 1998.

[8] R. Armellin, P. D. Lizia, F. B. Zazzera, K. Makino, and M. Berz, "Aphophis encounter 2029: Differential algebra and Taylor models," in *Planetary Defense Conference: Protecting Earth from Asteroids*, 2009.

[9] F. Zhao, "Automatic analysis and synthesis of controllers for dynamical systems based on phase-space knowledge." Ph.D. dissertation, MIT, 1998.

[10] A. Chutinan and B. Krogh, "Computing polyhedral approximations to flow pipes for dynamic systems," in *Proc. CDC'98*. IEEE, 1998.

[11] F. Benhamou and L. Granvilliers, "Continuous and interval constraints," in *Handbook of Constraint Programming*. Elsevier, 2006, pp. 571–590.

[12] C. Le Guernic and A. Girard, "Reachability analysis of hybrid systems using support functions," in *Proc. CAV'09*, ser. LNCS, vol. 5643. Springer, 2009, pp. 540–554.

[13] S. Sankaranarayanan, T. Dang, and F. Ivancic, "Symbolic model checking of hybrid systems using template polyhedra," in *Proc. TACAS'08*, ser. LNCS, vol. 4963. Springer, 2008, pp. 188–202.

[14] W. Kühn, "Zonotope dynamics in numerical quality control," in *Mathematical Visualization*, H.-C. Hege and K. Polthier, Eds. Heidelberg: Springer, 1998, pp. 125–134.

[15] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *Proc. HSCC'05*, ser. LNCS, vol. 3414. Springer, 2005, pp. 291–305.

[16] S. M. Furler, E. W. Kraegen, R. H. Smallwood, and D. J. Chisolm, "Blood glucose control by intermittent loop closure in the basal mode," *Diabetes Care*, vol. 8, pp. 553–561, 1985.

[17] M. E. Fisher, "A semiclosed-loop algorithm for the control of blood glucose levels in diabetics," *IEEE Trans. Biomed. Eng.*, vol. 38, no. 1, pp. 57–61, 1991.

[18] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*. SIAM, 2009.

[19] N. S. Nedialkov, K. R. Jackson, and G. F. Corliss, "Validated solutions of initial value problems for ordinary differential equations," *Applied Mathematics and Computation*, vol. 105, no. 1, pp. 21–68, 1999.

[20] N. S. Nedialkov, K. R. Jackson, and M. Neher, "On the blunting method in the verified integration of ODEs," in *Proc. the 6th Taylor Model Methods Workshop*, 2009.

[21] T. A. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi, "Beyond HYTECH: Hybrid systems analysis using interval numerical methods," in *Proc. HSCC'00*, ser. LNCS, vol. 1790. Springer, 2000, pp. 130–144.

[22] N. S. Nedialkov and M. von Mohrenschildt, "Rigorous simulation of hybrid dynamic systems with symbolic and interval methods," in *Proc. the American Control Conference*. IEEE, 2002, pp. 140–146.

[23] N. Ramdani and N. S. Nedialkov, "Computing reachable sets for uncertain nonlinear hybrid systems using interval constraint-propagation techniques," *Nonlinear Analysis: Hybrid Systems*, vol. 5, no. 2, pp. 149–162, 2011.

[24] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert, "Efficient solving of large non-linear arithmetic constraint systems with complex Boolean structure," *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 1, pp. 209–236, 2007.

[25] L. Granvilliers and F. Benhamou, "Algorithm 852: Realpaver: An interval solver using constraint satisfaction techniques," *ACM Trans. On Math. Software*, vol. 32, no. 1, pp. 138–156, 2006.

[26] S. Gao, M. K. Ganai, F. Ivancic, A. Gupta, S. Sankaranarayanan, and E. M. Clarke, "Integrating ICP and LRA solvers for deciding nonlinear real arithmetic problems," in *Proc. FMCAD'10*. IEEE, 2010, pp. 81–89.

[27] P. Nuzzo, A. Puggelli, S. A. Seshia, and A. L. Sangiovanni-Vincentelli, "CalCS: SMT solving for non-linear convex constraints," in *Proc. FMCAD'10*. IEEE, 2010, pp. 71–79.

[28] M. Althoff and B. H. Krogh, "Avoiding geometric intersection operations in reachability analysis of hybrid systems," in *Proc. HSCC'12*. ACM, 2012, pp. 45–54.

[29] S. Ratschan and Z. She, "Safety verification of hybrid systems by constraint propagation-based abstraction refinement," *ACM Trans. Embedded Comput. Syst.*, vol. 6, no. 1, 2007.

[30] L. Benvenuti, D. Bresolin, A. Casagrande, P. Collins, A. Ferrari, E. Mazzi, A. Sangiovanni-Vincentelli, and R. Villa, "Reachability computation for hybrid systems with Ariadne," in *Proc. the 17th IFAC World Congress*. IFAC Papers-OnLine, 2008.

[31] P. Prabhakar and M. Viswanathan, "A dynamic algorithm for approximate flow computations," in *Proc. HSCC'11*. ACM, 2011, pp. 133–142.

[32] J. D. Meiss, *Differential Dynamical Systems*. SIAM publishers, 2007.

[33] P. Cousot and R. Cousot, "Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints," in *Proc. POPL'77*, 1977, pp. 238–252.

[34] K. Makino and M. Berz, "Suppression of the wrapping effect by Taylor model-based verified integrators: Long-term stabilization by shrink wrapping," *J. Differential Equations and Applications*, vol. 10, no. 4, pp. 385–403, 2005.

[35] ——, "Suppression of the wrapping effect by Taylor model-based verified integrators: Long-term stabilization by preconditioning," *J. Differential Equations and Applications*, vol. 10, no. 4, pp. 353–384, 2005.

[36] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *Theor. Comput. Sci.*, vol. 138, no. 1, pp. 3–34, 1995.

[37] P. A. Parillo, "Semidefinite programming relaxation for semialgebraic problems," *Mathematical Programming Ser. B*, vol. 96, no. 2, pp. 293–320, 2003.

[38] G. M. Ziegler, *Lectures on Polytopes*, ser. Graduate Texts in Mathematics. Springer, 1995, vol. 152.

[39] T. Dang, C. Le Guernic, and O. Maler, "Computing reachable states for nonlinear biological models," in *Proc. CMSB'09*, ser. LNCS, vol. 5688. Springer, 2009, pp. 126–141.

[40] J. Sotomayor, L. F. Mello, and D. de Carvalho Braga, "Bifurcation analysis of the Watt governor system," *Comput. Appl. Math.*, vol. 26, no. 1, 2007.

[41] I. Mitchell and C. Tomlin, "Level set methods for computation in hybrid systems," in *Proc. HSCC'00*, ser. LNCS, vol. 1790. Springer, 2000, pp. 310–323.

[42] T. Dang, O. Maler, and R. Testylier, "Accurate hybridization of nonlinear systems," in *Proc. HSCC '10*. ACM, 2010, pp. 11–20.

[43] A. Platzer and E. M. Clarke, "Computing differential invariants of hybrid systems as fixedpoints," *FMSD*, vol. 35, no. 1, pp. 98–120, 2009.

[44] C. Tomlin, G. J. Pappas, and S. Sastry, "Conflict resolution for air traffic management: A study in multiagent hybrid systems," *IEEE Trans. Autom. Control*, vol. 43, no. 4, pp. 509–521, 1998.