FLOW* 1.2: More Effective to Play with Hybrid Systems

Xin Chen[1], Sriram Sankaranarayanan[2], and Erika Ábrahám[1]

1. RWTH Aachen University, Germany. {xin.chen, abraham}@cs.rwth-aachen.de
2. University of Colorado, Boulder, CO. srirams@colorado.edu

Abstract. This paper gives a brief overview of the new features introduced in the latest version of the tool FLOW*. We mainly describe the new efficient scheme for integrating linear ODEs. We show that it can efficiently handle the challenging benchmarks on which, to the best of our knowledge, only SpaceEx works. Moreover, it is also possible to extend the method to deal with unbounded initial sets. A comparison between FLOW* 1.2 and SpaceEx on those benchmarks is given. Besides, we also investigate the scalability FLOW* 1.2 based on our non-linear line circuit benchmarks.

1 Introduction

Nowadays *Cyber-Physical Systems (CPSs)* are ubiquitous in various areas such as automotive, biology, medicine, electrical engineering and etc.. A wide class of CPS are hybrid systems which exhibit both continuous flow and discrete jumps. Since hybrid systems are often safety-critical, we are interested in answering the problem of whether a hybrid system can reach an unsafe state or not. To do that, we need to explore the state space of the system, and to determine whether a state or a region is reachable. It is already known that there is no *decision procedure* to answer the reachability problem for general hybrid systems [1], people thereby resort to approximation techniques.

In safety verification, we could compute a superset for the exact reachable set. If there is no unsafe state included, then the system is safe. In the past two decades, a great amount of work was devoted to investigating new techniques for over-approximating the reachable sets of hybrid systems whose dynamics are defined by linear expressions. Such systems are also called *linear hybrid systems*. The popular over-approximate representations are convex polytopes [9], ellipsoids [18], zonotopes [14] and support functions [19]. Along with their computation techniques, the tools CheckMate [23], Ellipsoidal Toolbox, PHAVer [11] and SpaceEx [12] are developed. On the other hand, few reachable set representations other than intervals (boxes) [21] are successfully applied to non-linear hybrid systems. The typical tools are Ariadne [2], iSAT [10], dReach [13] and HyCreate [17]. Since interval representation suffers from bad scalability, dealing with more than 4 system variables is still a difficult task in general.

In [4], we proposed a method of using Taylor models as the over-approximate representation for a reachable set segment which is also called a *flowpipe*. It has a good performance on both linear and non-linear hybrid systems. We later released a tool named FLOW* [5] which mainly focuses on safety verification of non-linear hybrid systems. Our experimental results show that the technique provides a very promising way of using higher-order over-approximations for hybrid system reachable sets.

In this paper, we give a brief introduction to the new enhancement of FLOW* in the version 1.2. It mainly consists of (1) an efficient scheme of computing Taylor model flowpipes for linear Ordinary Differential Equations (ODEs), (2) scalability improvement, and (3) improving the intersection and aggregation algorithm. We will address (1) and (2) in the rest of the paper.

2 Preliminaries

In this section, we give the basic definitions of interval and Taylor model arithmetic, the standard Taylor model integration algorithm is also revisited.

Interval arithmetic. A (closed and bounded) *interval* is represented in the form of $[a, b]$ wherein a, b are rational numbers and $a \leq b$. It defines the set of reals between a and b. The operations on reals can be extended to handling intervals. For example, $[a, b] + [c, d] = [a + c, b + d]$, and $[a, b] \cdot [c, d] = [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}]$. Intervals can also be organized as vectors or matrices. Given an interval vector (or matrix resp.) V, we have that $v \in V$ for a real vector (or matrix resp.) v iff each entries of v is contained in the corresponding entries of V.

Taylor models are introduced as over-approximate representations for continuous functions which are not necessarily of closed form. A *Taylor Model (TM)* is denoted as a pair (p, I) such that p is a polynomial over a finite set of variables each of which ranges in an interval, while I is an interval which is an enclosure of the remainder part.

Given a continuous function $f(\boldsymbol{x})$ with $\boldsymbol{x} \in D$, we say that f is over-approximated by a TM (p, I), denoted by $f \in (p, I)$, iff $f(\boldsymbol{x}) \in p(\boldsymbol{x}) + I$ for all $\boldsymbol{x} \in D$. TMs can also be organized as vectors to over-approximate vector-valued functions. Here, we also call them TMs. To recognize that a TM $(p(\boldsymbol{x}), I)$ with $\boldsymbol{x} \in D$ defines a *convex set* of continuous functions over D, we pick any functions $f, g \in (p, I)$, then all continuous functions between f, g also belong to (p, I), i.e., for any h such that $h(\boldsymbol{x})$ is between $f(\boldsymbol{x}), g(\boldsymbol{x})$ for all $\boldsymbol{x} \in D$ is also in (p, I). For simplicity, we call (p, I) a TM of f when $f \in (p, I)$.

Taylor model arithmetic. Given two continuous functions f, g as well as their TMs (p_1, I_1), (p_2, I_2) respectively. A TM for $f + g$ can be computed by adding the polynomial and remainder parts respectively, i.e., $(p_1 + p_2, I_1 + I_2)$, while an order k TM for their product $f \cdot g$ can be computed by

$$(\, p_1 \cdot p_2 - r_k \, , \; I_1 \cdot B(P_2) + I_2 \cdot B(P_1) + I_1 {\cdot} I_2 + B(r_k) \,)$$

wherein $B(p)$ denotes an interval enclosure of the range of p, and the *truncated part* r_k consists of the terms in $p_1 \cdot p_2$ of degrees $> k$. More operations are defined in [20].

Taylor model integration. Given an ODE $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, t)$, wherein f is at least *locally Lipschitz continuous* w.r.t. $\boldsymbol{x}$, and a TM initial set X_0, we want to compute an over-approximation of the solutions from X_0 within the bounded time interval $[0, \Delta]$. It is carried out by computing a set of TMs $\mathcal{F}_1, \ldots, \mathcal{F}_N$ which over-approximate the reachable set over the time intervals $[0, \delta], \ldots, [(N - 1)\delta, N\delta]$, wherein $\Delta = N\delta$, respectively. The value of δ is called a *step-size*, and it is also not necessary to use a uniform step-size for all TM flowpipes. The algorithm of computing TM flowpipes are given as follows.

```
1: for $i = 1, \ldots, N$ do
2:     Compute a Taylor expansion $p_i(\boldsymbol{x}_i, t)$ for the ODE solution from $\boldsymbol{x}_i$.
3:     Evaluate an interval remainder $I_i$ such that $(p_i, I_i)$ is a TM for the ODE solution
       when $\boldsymbol{x}_i \in X_{i-1}$ and $t \in [0, \delta]$.
4:     Compute a TM $X_i$ for $(p_i(X_{i-1}, \delta), I_i)$.
5: end for
```

To find a proper remainder interval, we start with a user given estimation I and check the contractiveness of the Picard operation

$$\mathbb{P}_f(g)(\boldsymbol{x}_0, t) = \boldsymbol{x}_0 + \int_0^t f(g(\boldsymbol{x}_0, s), s) ds$$

over the TM (p_i, I) by TM arithmetic. If the Picard operation is contractive, then by *Schauder fixed point theorem*, a fixed point which is the unique solution is contained in (p_i, I). Such a remainder can be further refined by repeatedly applying Picard operation.

The most time-costly step in the above algorithm is the remainder evaluation, and we will see in the next section that it can be replaced by a more efficient method.

3 Efficient flowpipe construction for linear ODEs

Without loss of Generality, we consider the linear ODE of the form $\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{u}$ such that A is a constant square matrix and $\boldsymbol{u}$ is a time-varying uncertainty whose range is defined by a bounded interval $\mathcal{U}$. The flowpipe construction for such ODEs has already been extensively studied, several representations such as zonotopes [14] and support functions [19] are shown to be effective to deal with large scale cases. Here, we describe an efficient Taylor model-based method which also has a good scalability. It is a combination of the methods given in [15] and [22].

The linear ODE has the following closed form solution w.r.t. an initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$.

$$\varphi(\boldsymbol{x}_0, t) \; = \; e^{At}\boldsymbol{x}_0 \; + \; \int_0^t e^{A(t-s)}\boldsymbol{u} \, ds \tag{1}$$

However, the matrix e^{At} is hard to approximate when the time t is sufficiently large. Therefore, it is more convenient to use the following recurrence relation to generate flowpipe over-approximations,

$$\mathcal{F}_i = \Phi \mathcal{F}_{i-1} \oplus \mathcal{B} \qquad \text{for } i = 2, 3, \ldots, N \tag{2}$$

such that $\mathcal{F}_1$ is an over-approximation of the first flowpipe, i.e., the set $\{\varphi(\boldsymbol{x}_0, t) \,|\, t \in [0, \delta]\}$, $\oplus$ denotes Minkowski sum[1], $\Phi = e^{A\delta}$, and $\mathcal{B} = \{\int_0^\delta e^{A(\delta - s)}\boldsymbol{u}(s) \, ds \,|\, \boldsymbol{u}(s) \in \mathcal{U}\}$. Then $\mathcal{F}_i$ is an over-approximation of $\{\varphi(\boldsymbol{x}_0, t) \,|\, t \in [(i-1)\delta, i\delta]\}$ for $i = 2, \ldots, N$. Then, we show how to compute TM flowpipes based on the above recurrence relation.

[1] $X \oplus Y = \{x + y \,|\, x \in X, y \in Y\}$.

Compute the first flowpipe. The purpose is to compute a TM $(p_1(\boldsymbol{x}_0,t),I_1)$ such that $\varphi(\boldsymbol{x}_0,t)\in p_1(\boldsymbol{x}_0,t)+I_1$ for all $t\in[0,\delta]$. To do so, we first compute an order k TM matrix over-approximation $(p_\Phi(t),I_\Phi)$ for e^{At},

$$p_\Phi(t) \;=\mathcal{I}+At+\frac{1}{2}A^2t^2+\cdots+\frac{1}{k!}A^kt^k$$

wherein $\mathcal{I}$ is an identity matrix. The remainder interval matrix I_Φ can be evaluated base on the Lagrange remainder $\frac{A^{k+1}t^{k+1}}{(k+1)!}e^{A\xi}$ with some $\xi\in[0,\delta]$. That is, we first compute the value of $\rho=e^{|A|\delta}$ wherein $|\cdot|$ denotes the maximum norm. Then the matrix $e^{A\xi}$ is contained in the interval matrix M_ρ whose entries are all defined by $[-\rho,\rho]$. Hence, $I_\Phi=\frac{A^{k+1}t^{k+1}}{(k+1)!}M_\rho$ contains the remainder for p_Φ. By choosing k sufficiently large, we are able to obtain arbitrarily good accuracy.

The TM for $e^{A(t-s)}$ can be computed similarly, and then by using TM arithmetic, we are able to obtain a TM $(p_\mathcal{B}(t),I_\mathcal{B})$ for $\int_0^t e^{A(t-s)}\mathcal{U}\,ds$. Hence, the first TM flowpipe $\mathcal{F}_1$ is the result of computing $(p_\Phi(t),I_\Phi)\cdot\boldsymbol{x}_0+(p_\mathcal{B}(t),I_\mathcal{B})$.

Lemma 1. *For all $t\in[0,\delta]$, we have that $\varphi(\boldsymbol{x}_0,t)\in(p_\Phi(t),I_\Phi)\cdot\boldsymbol{x}_0+(p_\mathcal{B}(t),I_\mathcal{B})$.*

Compute the remaining flowpipes. By expanding the recurrence relation (2), the i-th TM flowpipe can also be computed as

$$\mathcal{F}_i=\Phi^{i-1}\mathcal{F}_1\oplus\bigoplus_{j=0}^{i-2}\Phi^j\mathcal{B}$$

In our case, we represent Φ by the interval matrix $(p_\Phi(\delta),I_\Phi)$ and $\mathcal{B}$ by the interval vector $(p_\mathcal{B}(\delta),I_\mathcal{B})$. Hence, we can use TM arithmetic to compute

$$\begin{aligned}\mathcal{F}_i(t)\;&=(p_\Phi(\delta),I_\Phi)^{i-1}\cdot(p_\Phi(t),I_\Phi)\cdot\boldsymbol{x}_0+(p_\Phi(\delta),I_\Phi)^{i-1}\cdot(p_\mathcal{B}(t),I_\mathcal{B})\\&\quad+\sum_{j=0}^{i-2}\big((p_\Phi(\delta),I_\Phi)^j\cdot(p_\mathcal{B}(\delta),I_\mathcal{B})\big)\end{aligned}\tag{3}$$

wherein $t\in[0,\delta]$. Since I_Φ can be made arbitrarily small when the order k is large enough, the main source of wrapping effect is $I_\mathcal{B}$ if we use TM arithmetic to evaluate $\mathcal{F}_i$. To avoid that, we use support function to represent $I_\mathcal{B}$ during the computation, and then obtain a tight remainder interval for $\mathcal{F}_i$.

Theorem 1. *For all $1\le i\le N$, we have that $\varphi(\boldsymbol{x}_0,(i-1)\delta+t)\in\mathcal{F}_i(t)$ for $t\in[0,\delta]$.*

Unbounded initial sets. The above flowpipe construction method can be easily extended to handle unbounded initial sets which are defined by polynomial constraints. Although we only describe the TM over-approximation method for a forward flowmap, a backward one can be obtained in a similar way, and the approximation quality will still only depend on the step-size and the TM order. Then, given an initial set X_0 defined by a system of polynomial constraints, we are able to compute both over- and under-approximations for the reachable sets by propagating the constraints of X_0 via the backward flowmap over-approximation. The details are described in [6].

			FLOW*				SpaceEx (LGG)				SpaceEx (STC)		
benchmark	var	T	δ	k	P	time	δ	tolerance abs.	box	octagon	tolerance rel.	box	octagon
filtered oscillator 6	6	$[0, 4]$	0.05	8	128	2.4	0.05	0.01	0.2	3.4	0.05	0.4	6.7
filtered oscillator 10	10	$[0, 4]$	0.05	8	128	6.1	0.05	0.01	0.5	55	0.05	1.2	36
filtered oscillator 18	18	$[0, 4]$	0.05	8	128	22	0.05	0.01	1.1	815	0.05	3.0	372
filtered oscillator 34	34	$[0, 4]$	0.05	8	128	106	0.05	0.01	3.6	T.O.	0.05	8.6	T.O.
helicopter	29	$[0, 30]$	0.2	70	256	198	0.1	0.001	35	T.O.	0.05	12	340

Table 1. FLOW* v.s. SpaceEx on the SpaceEx benchmarks. Legends: var: # of variables, T: time horizon, δ: time step-size, k: TM order, P: precision, box: box over-approximation, octagon: octagon over-approximation, T.O.: $>$ 2000 seconds (time out).

4 Experimental results

4.1 Comparison with SpaceEx

We present a comparison between FLOW* 1.2 and SpaceEx 0.98 over the benchmarks which are contained in the package of SpaceEx. Since FLOW* and SpaceEx use different representations for flowpipes, a fair overall comparison on their accuracy is not easy. Thereby we choose the computation settings for both of the tools such that the last box flowpipe computed by SpaceEx contains the TM flowpipe(s) over the same time interval. We give the experimental results in Table 1. We consider both of the LGG and STC algorithms implemented in SpaceEx. It seems that the advantage of STC becomes clearer when the system scale grows.

Since it is difficult to exactly implement support functions, SpaceEx computes boxes or octagons as further over-approximations. For a d-dimensional system, a box over-approximation requires to compute d samples on the boundary of a support function according to the box template, whereas an octagon one requires to compute $O(d^2)$ many. Although it is often unnecessary to compute octagon over-approximations according to the critical directions given by users or unsafe sets, we sometimes still need to do that for reusing the flowpipes. On the other hand, TM flowpipes are already overall accurate and can be directly reused with different unsafe specifications.

4.2 Scalability evaluation

Scalability is very significant in the applicability of a tool. Here, we investigate the scalability of FLOW* 1.2 based on a non-linear resistor circuit benchmark shown in Figure 1. The model is originally studied by Chen et al. [8], and then adapted to be interesting hybrid case studies [16]. The circuit is composed of $n+1$ non-linear resistors and the same number of capacitors. Each non-linear resistor consists of a diode and a unit resistor ($r = 1$). For simplicity, we assume that all capacitors have unit capacitance

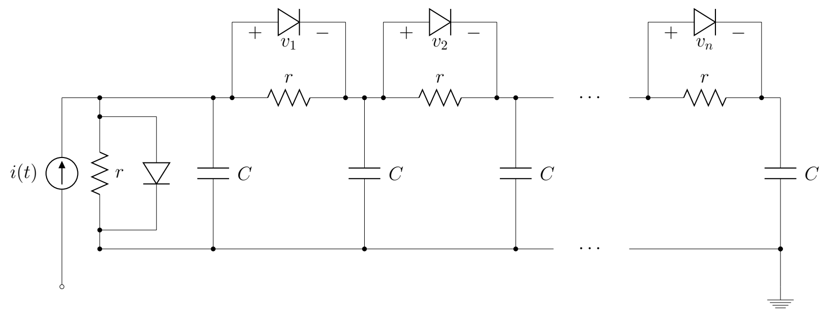

Fig. 1. Transmission line circuit

$C = 1$. For each diode, the I-V characteristic is given by $I = e^{\alpha \cdot V} - 1$. The current source $i(t)$ in the figure is the input, and v_1 is the single output of the circuit. Therefore, the whole circuit system can be described by the following ODE.

$$\begin{cases} \dot{v}_1 &= -2 \cdot v_1 + v_2 + 2 - e^{\alpha \cdot v_1} - e^{\alpha \cdot (v_1 - v_2)} + i(t) \\ \dot{v}_2 &= -2 \cdot v_2 + v_1 + v_3 + e^{\alpha \cdot (v_1 - v_2)} - e^{\alpha \cdot (v_2 - v_3)} \\ & \cdots \\ \dot{v}_{n-1} &= -2 \cdot v_{n-1} + v_{n-2} + v_n + e^{\alpha \cdot (v_{n-2} - v_{n-1})} - e^{\alpha \cdot (v_{n-1} - v_n)} \\ \dot{v}_n &= -v_n + v_{n-1} - 1 + e^{\alpha \cdot (v_{n-1} - v_n)} \end{cases}$$

In order to avoid the stiffness in the original model, we reduce the value of α from 40 to 5, but it is still challenging.

Scalable continuous and hybrid benchmarks can be built based on various types of inputs. We consider the following continuous and hybrid inputs.

$$\text{continuous:} \quad i(t) = \sin(5t), \qquad \text{hybrid:} \quad i(t) = \begin{cases} 2, & t \le 1 \\ 3 - t, & 1 < t \le 2 \\ 1, & t > 2 \end{cases}$$

The running time of FLOW* on different scales are listed in Table 2. More case studies of continuous and hybrid systems can be found in our benchmark collection [7].

5 How to make the best use of FLOW*?

It is always tricky to choose a proper computational setting for a reachability problem in FLOW*. We plan to give a short guide in this section. Except the scheme introduced in Section 3 and the one for handling non-polynomial ODEs, FLOW* 1.2 provides 3 different integration schemes for polynomial ODEs. Table 2 summarizes the suggested situations for applying them. The details are described elsewhere [3].

scheme	var	degree	TM order
poly ode 1	≤ 3	≤ 5	≤ 5
poly ode 2	≤ 5	≤ 5	≥ 6
poly ode 3	any	≥ 6	any

Fig. 2. Suggested situations for applying the integration schemes

	continuous					hybrid				
n	δ	k	I_e	ε	t (s)	δ	k	I_e	ε	t (s)
2	0.03	$3 \sim 6$	$[-10^{-3}, 10^{-3}]$	10^{-12}	1.4	0.01	$3 \sim 6$	$[-10^{-3}, 10^{-3}]$	10^{-12}	2.3
4	0.01	$3 \sim 6$	$[-10^{-5}, 10^{-5}]$	10^{-10}	56	0.01	$3 \sim 6$	$[-10^{-4}, 10^{-4}]$	10^{-10}	48
6	0.0002 ~ 0.02	4	$[-10^{-5}, 10^{-5}]$	10^{-8}	73	0.0002 ~ 0.02	4	$[-10^{-5}, 10^{-5}]$	10^{-8}	243
8	0.0002 ~ 0.01	4	$[-10^{-5}, 10^{-5}]$	10^{-8}	176	0.0002 ~ 0.01	4	$[-10^{-5}, 10^{-5}]$	10^{-8}	851
10	0.0002 ~ 0.005	4	$[-10^{-5}, 10^{-5}]$	10^{-7}	205	0.0002 ~ 0.005	4	$[-10^{-5}, 10^{-5}]$	10^{-7}	904
12	0.0002 ~ 0.005	4	$[-10^{-5}, 10^{-5}]$	10^{-7}	402	0.0002 ~ 0.005	4	$[-10^{-5}, 10^{-5}]$	10^{-7}	1933

Table 2. Scalability evaluation of FLOW* on the non-linear line circuit benchmarks. Legends: δ: time step-size, k: TM order, I_e: remainder estimation, ε: cutoff threshold.

It is always tricky to choose the parameters for FLOW* to perform a reachability analysis. An improper setting may easily generate a bad experimental result for a benchmark, but it does not mean the incapability of FLOW*. We have the following suggestions for users.

(i) Since the purpose of using TMs is to provide higher-order over-approximations, a better performance can usually be obtained by using high TM orders and large step-sizes than the other way around. When we need to improve the approximation quality of a result, a higher TM order is often preferred to a smaller step-size.
(ii) When a system has more than 6 variables, computing high order TMs may cost too long time. If that is the case, we suggest to increase the cutoff threshold such that polynomials will be simplified by moving the small terms into remainders.
(iii) The reason to use adaptive techniques is to reduce computation time by omitting "trivially small" overestimation. Hence, they work well only in the case that the given remainder estimations are small enough.

6 Conclusion and future work

In the paper, we gave a brief overview of the new enhancement in FLOW*. The performance on linear systems is greatly improved and competitive to SpaceEx. In the future, we plan to extend the functionality of the tool in the following aspects: (a) Scalable computation of both over- and under-approximations for linear continuous systems, such that the initial sets are defined by polynomial constraints; (b) Computing TM flowpipes for time-delay systems; (c) A better scheme to deal with stiff dynamics.

References

1. 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.*, 138(1):3–34, 1995.

2. 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. of the 17th IFAC World Congress*. IFAC Papers-OnLine, 2008.
3. X. Chen. *Reachability Analysis of Non-Linear Hybrid Systems Using Taylor Models*. PhD thesis, RWTH Aachen University, 2015.
4. X. Chen, E. Ábrahám, and S. Sankaranarayanan. Taylor model flowpipe construction for non-linear hybrid systems. In *Proc. RTSS'12*, pages 183–192. IEEE, 2012.
5. X. Chen, E. Ábrahám, and S. Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In *Proc. of CAV'13*, volume 8044 of *LNCS*, pages 258–263. Springer, 2013.
6. X. Chen, S. Sankaranarayanan, and E. Ábrahám. Under-approximate flowpipes for non-linear continuous systems. In *Proc. FMCAD'14*, pages 59–66, 2014.
7. X. Chen, S. Schupp, I. Ben Makhlouf, E. Ábrahám, G. Frehse, and S. Kowalewski. A benchmark suite for hybrid systems reachability analysis. *To appear in NFM'15*, 2015.
8. Y. Chen and J. White. A quadratic method for nonlinear model order reduction. In *Proc. MSM*, pages 477–480, 2000.
9. A. Chutinan and B. Krogh. Computing polyhedral approximations to flow pipes for dynamic systems. In *Proc. CDC'98*. IEEE, 1998.
10. 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*, 1:209–236, 2007.
11. G. Frehse. PHAVer: Algorithmic verification of hybrid systems past HyTech. In *Proc. HSCC'05*, volume 2289 of *LNCS*, pages 258–273. Springer, 2005.
12. 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*, volume 6806 of *LNCS*, pages 379–395. Springer, 2011.
13. S. Gao. *Computable Analysis, Decision Procedures, and Hybrid Automata: A New Framework for the Formal Verification of Cyber-Physical Systems*. PhD thesis, Carnegie Mellon University, 2012.
14. A. Girard. Reachability of uncertain linear systems using zonotopes. In *Proc. HSCC'05*, volume 3414 of *LNCS*, pages 291–305. Springer, 2005.
15. A. Girard, C. Le Guernic, and O. Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. In *Proc. HSCC'06*, volume 3927 of *LNCS*, pages 257–271. Springer, 2006.
16. C. Gu. *Model Order Reduction of Nonlinear Dynamical Systems*. PhD thesis, University of California, Berkeley, 2011.
17. HyCreate: A tool for overapproximating reachability of hybrid automata.
18. A. A. Kurzhanskiy and P. Varaiya. Ellipsoidal toolbox. Technical Report UCB/EECS-2006-46, EECS Department, University of California, Berkeley, May 2006.
19. C. Le Guernic. *Reachability Analysis of Hybrid Systems with Linear Continuous Dynamics*. PhD thesis, Université Joseph Fourier, 2009.
20. K. Makino and M. Berz. Taylor models and other validated functional inclusion methods. *International Journal of Pure and Applied Mathematics*, 4(4):379–456, 2003.
21. N. Ramdani and N. S. Nedialkov. Computing reachable sets for uncertain nonlinear hybrid systems using interval constraint-propagation techniques. *Nonlinear Analysis: Hybrid Systems*, 5(2):149–162, 2011.
22. S. Sankaranarayanan, T. Dang, and F. Ivancic. Symbolic model checking of hybrid systems using template polyhedra. In *Proc. TACAS'08*, volume 4963 of *LNCS*, pages 188–202. Springer, 2008.
23. B. I. Silva, K. Richeson, B. H. Krogh, and A. Chutinan. Modeling and verification of hybrid dynamical system using CheckMate. In *Proc. ADPM*, 2000.