An Algorithmic Approach to Stability Verification of Hybrid Systems: A Summary

Pavithra Prabhakar and Miriam García Soto

IMDEA Software Institute
28223 Pozuelo de Alarcón, Madrid
pavithra.prabhakar@imdea.org
miriam.garcia@imdea.org

Abstract

This paper summarizes results related to a novel algorithmic approach for verifying stability of hybrid systems. The traditional approach based on Lyapunov function search suffers from several disadvantages — it relies on the user expertise to obtain good templates for the Lyapunov function; further, an unsuccessful attempt at instantiating the templates provides no insights into the choice of better templates. To overcome these difficulties, the algorithmic approach relies on an abstraction refinement framework which systematically searches for a proof and provides insights to the user in the event of a failure to prove stability. We summarize the new foundations, techniques and software tools that we have developed for the algorithmic approach to stability verification.

Contents

1 Introduction

Stability is a fundamental property in control system design, which captures the notion that small perturbations in the initial state of the system result in only small perturbations in the behaviors starting from than point. Traditional methods in control theory for stability analysis of dynamical systems consist in exhibiting a certificate of stability in the form of a Lyapunov function — a continuously differentiable, positive definite function such that the value of the function along any solution of the system is decreasing. This idea is extended to the case of hybrid systems — systems exhibiting mixed discrete continuous behaviors — through the notions of common Lyapunov functions and multiple Lyapunov functions. In terms of automation, the Lyapunov function search is carried out by fixing a template and encoding the conditions of the Lyapunov function in some constraint solving formalism such as Linear Matrix Inequalities (LMIs) or Sum-of-Squares (SOS) optimization. The main difficulty with this approach is determining the right template for the candidate Lyapunov function. Further, when the constraint solvers fails to obtain parameters for the template, they do not provide any insights into the reason for instability or for the choice of better templates for subsequent trials. Motivated by these issues, in this line of work, we investigate an algorithmic approach based on abstraction refinement, which systematically explore the search space, and returns counter-examples which indicate potential reasons for instability and guide the choice of subsequent abstractions.

2 Hybrid system

We present a semantic model for representing the mixed discrete-continuous behaviors of a hybrid system [3, 6]. A *hybrid system* $\mathcal{H}$ is a tuple (Q, X, Σ, Δ), where:

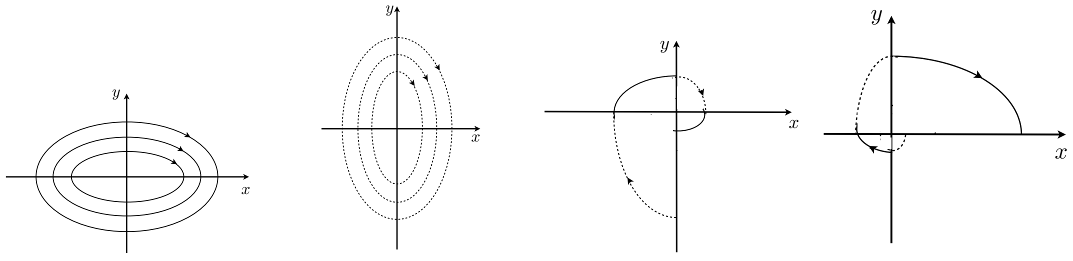
- Q is a finite set of control locations;
- $X = \mathbb{R}^n$, for some n, is the continuous state space; $Q \times X$ is the state-space;
- $\Sigma \subseteq \mathit{Trans}(Q, X)$ is a set of transitions, where $\mathit{Trans}(Q, X) = (Q \times X) \times (Q \times X)$; and
- $\Delta \subseteq \mathit{Traj}(Q, X)$ is a set of trajectories, where $\mathit{Traj}(Q, X)$ is the set of all functions $\tau : I \to Q \times X$, where I is either a finite $[0, T]$ or infinite $[0, \infty)$ time interval, such that the function restricted to the discrete space, Q, is finitely varying and restricted to the continuous space, is a continuous function.

Σ captures the switching conditions and resets during the mode switch and Δ captures the solutions of differential equations or inclusions modelling the continuous dynamics. Given a state $s = (q, x)$ we use the subscript X to denote the continuous part of the state, that is, $s_X = x$. An *execution* of a hybrid system $\mathcal{H}$ is a sequence of transitions and trajectories $\sigma : D \to \Sigma \cup \Delta$, where D is either $\{0, \ldots, n\}$ for some n or $\mathbb{N}$. We denote the set of all executions of the system by $\mathit{Exec}(\mathcal{H})$.

Example **1.** *In Figure 1, we depict an example of a particular class of hybrid systems, namely, linear switched systems. The continuous evolution of this system is determined by linear dynamical systems. Let us consider two linear dynamical systems defined by matrices A and B. The phase portraits for both systems are shown in Figure 1(a) and Figure 1(b) respectively. We define two linear switched systems $\mathcal{M}_1$ and $\mathcal{M}_2$, by switching between these linear dynamical systems. $\mathcal{M}_1$ follows the dynamics determined by matrix B in the first and third quadrants while follows A in the other two quadrants. $\mathcal{M}_2$ follows A in the first and third quadrants and follows B in the second and fourth quadrants. Sample executions for both systems are shown in Figure 1(c) and Figure 1(d). These executions consists of trajectories, which capture the evolution of the system in a particular quadrant, and switching transitions, that occur at the boundaries of the quadrants.*

3 Lyapunov and Asymptotic Stability

In this section, we define two classical notions of stability for hybrid systems, namely, Lyapunov and asymptotic stability. We consider stability with respect to an equilibrium point. We consider, for simplicity, the equilibrium point to be the origin, which is denoted as $0_{\mathcal{H}}$. Lyapunov



(a) $\dot{x} = Ax$ (b) $\dot{x} = Bx$ (c) System $\mathcal{M}_1$ (d) System $\mathcal{M}_2$

Figure 1: Phase portraits and sample executions

stability captures the notion that small perturbations in the initial state of the system result in only small perturbations of the behaviors starting from that point. Asymptotic stability requires convergence in addition to Lyapunov stability.

Definition 1. *A set of executions $S \subseteq$ Exec$(\mathcal{H})$ is said to be* Lyapunov stable *if for every $\epsilon > 0$, there exists a $\delta > 0$ such that for every execution $\sigma \in S$ with $\sigma(0)(0)_X \in \mathcal{B}_\delta(0_\mathcal{H})$, $\sigma(i)(t)_X \in \mathcal{B}_\epsilon(0_\mathcal{H})$ for all i, t. A hybrid system $\mathcal{H}$ is said to be Lyapunov stable, if Exec$(\mathcal{H})$ is Lyapunov stable.*

An execution σ of $\mathcal{H}$ is said to *converge* to 0, denoted *Conv*$(\sigma, 0_\mathcal{H})$, if for every $\epsilon > 0$, there exists a time $T \in \mathbb{R}_{\geq 0}$ such that $\sigma(i)(t)_X \in \mathcal{B}_\epsilon(0_\mathcal{H})$ for every $t \geq T$.

Definition 2. *A set of executions $S \subseteq$ Exec$(\mathcal{H})$ is said to be* asymptotically stable *if it is Lyapunov stable and there exists a $\delta > 0$ such that every $\sigma \in S$ with $\sigma(0)(0)_X \in \mathcal{B}_\delta(0_\mathcal{H})$, Conv$(\sigma, 0_\mathcal{H})$ holds. A hybrid system $\mathcal{H}$ is said to be asymptotically stable if Exec$(\mathcal{H})$ is asymptotically stable.*

Example **2.** *We observe in Figure 1(c) that executions from system $\mathcal{M}_1$ converge to the origin, which implies asymptotic stability. In the case of system $\mathcal{M}_2$, the execution of Figure 1(d) diverges, hence we know the system is unstable.*

4 The Algorithmic Approach

Our broad approach is to develop an abstraction refinement framework for stability verification. In this section, we define the foundations and the algorithmic approach based on abstractions for stability analysis.

4.1 Preorders and equivalences

Simulation and bisimulation relations are the classical notions of pre-order and equivalence on systems which preserve several discrete-time properties such as safety and those expressible in linear and branching time logics. However, it turns out that they do not preserve stability [8]. Hence, a stronger notion that imposes certain continuity constraints on the relations — continuous simulations and bisimulations — is proposed to enforce stability preservation.

Theorem 4.1. *[8, 12] Let R be a continuous simulation from a hybrid system $\mathcal{H}_1$ to a hybrid system $\mathcal{H}_2$. Then:*

- *$\mathcal{H}_2$ is Lyapunov stable implies $\mathcal{H}_1$ is Lyapunov stable.*
- *$\mathcal{H}_2$ is asymptotically stable implies $\mathcal{H}_1$ is asymptotically stable.*

The above results suggest that the standard constructions of abstractions and minimizations for analysis of properties such as safety which rely on simulations and bisimulations may not carry over directly for stability analysis. Next, we present a "quantitative" version of the standard predicate abstraction for stability analysis.

4.2 Quantitative Predicate Abstraction (QPA)

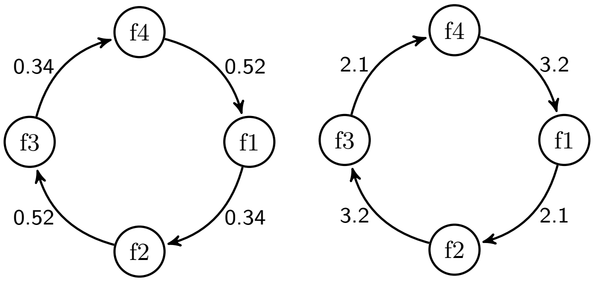
In the context of safety verification, a finite abstraction of a concrete system is constructed from a partition of the state space of the system into a finite number of regions. The nodes in the finite abstraction correspond to the regions, and the edges between two nodes capture the existence of an execution in the concrete system starting from a state in the region corresponding to the first node to a state in the region corresponding to the second node. This defines an abstract system, a finite graph, which over-approximates the behaviors of the concrete system, and hence, safety of the abstract system implies the safety of the concrete system.

However, for stability verification, it does not suffice to merely construct a system which over-approximates the behaviors of the concrete system. We need to capture some quantitative information about the evolution of the distance of the states to the origin along an execution. Hence, we annotate the finite graph with weights. More precisely, we interpret the nodes in the abstract graph as regions, an edge in the graph as the existence of a potential execution from one region to other evolving through a third region, and the weights as the scaling in the distance to the origin of the execution as it traverses from the first region to the second one. The crux of this construction lies in computing a predicate $\mathit{ReachRel}_{P_1,P_2}(s_1, s_2)$, where P_1 and P_2 are regions in the state-space partition. The predicate holds for a pair of states (s_1, s_2) if $s_1 \in P_1$, $s_2 \in P_2$ and there exists a region P in the partition and an execution of the system from s_1 to s_2 which always remains in P except for the initial and the final times. If $\mathit{ReachRel}_{P_1,P_2}$ evaluates to true for some (s_1, s_2), then there is an edge from P_1 to P_2. The weight on the edge is an upper bound on the "scaling", that is, the ratio $\frac{||s_2||}{||s_1||}$, over all (s_1, s_2) which satisfy $\mathit{ReachRel}_{P_1,P_2}$. In addition, we annotate a node with the label "conv" if all the executions which eventually remain in the region converge to the origin.

Remark 1. *In practice, to succeed in proving stability, we may need to prune the graph to remove redundant information. For instance, [9, 10] consider only the facets of the regions of the partition as nodes.*

Example **3.** *We illustrate the QPA construction on the linear switched systems in Example 1, $\mathcal{M}_1$ and $\mathcal{M}_2$. We present a simple construction of the weighted graph. The nodes corresponding to the quadrants and origin are not considered because they add redundant information on executions of the system.*

Figure 2 shows the finite weighted graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, corresponding to the systems $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. The construction of $\mathcal{G}_1$ and $\mathcal{G}_2$ is performed by partitioning the planar space into four quadrants, and choosing as the nodes the boundaries to the quadrants — the positive x axis f_1, the negative x axis f_3, the positive y axis f_2 and the negative y axis f_4. We observe



(a) Weighted graph $\mathcal{G}_1$ (b) Weighted graph $\mathcal{G}_2$

Figure 2: Quantitative Abstractions

in Figure 1(c) and Figure 1(d) that an execution evolves in the clockwise direction reflected by the edge directions. The weights define an upper bound on the scalings, for instance, a weight 0.52 *on the edge* $f_4 f_1$ *indicates that the execution starting at distance* d *on the positive* y *axis, when it crosses the first quadrant and reaches the positive* x *axis is at most at distance* $0.52d$ *from the origin.*

4.3 Model-Checking Algorithm and Soundness of QPA

Next, we present the model-checking algorithm which is run on the abstract weighted graph, and provide insights into the correctness of the same. We say that a hybrid system $\mathcal{H}$ is well-behaved with respect to a partition of the state-space $\mathcal{H}$ if every trajectory of $\mathcal{H}$ with finite time domain is finitely varying with respect to the partition, that is, enters and exist any region only finitely many time.

Theorem 4.2. *[12] Let* $\mathcal{G}$ *be a quantitative abstraction of a hybrid system* $\mathcal{H}$ *with respect to a partition* P*, and let* $\mathcal{H}$ *be well-behaved with respect to* P*. Consider the following conditions:*

G1 There is no edge e *in* $\mathcal{G}$ *with infinite weight.*

G2 The product of the weights on every simple cycle π *of* $\mathcal{G}$ *is less than or equal to* 1*.*

G3 Every node in $\mathcal{G}$ *is labelled by "conv".*

G4 The product of the weights on every simple cycle π *of* $\mathcal{G}$ *is strictly less than* 1*.*

Then:

- $\mathcal{H}$ *is Lyapunov stable if conditions G1 and G2 hold; and*
- $\mathcal{H}$ *is asymptotically stable if conditions G3 and G4 hold.*

Every execution of the hybrid system corresponds to a path in the weighted graph. If the execution switches infinitely between the regions (note that an execution is allowed to switch infinitely between the regions, but not a trajectory), then it corresponds to an infinite path in the graph. The product of the weights on prefixes of this path provide an upper bound on the scaling associated with the corresponding prefix of the execution. Note that any finite path can be decomposed into a simple path and a set of simple cycles, hence, if condition $G2$ holds the weight of the path is upper bounded by the weight of any simple path in the graph. Therefore, there is global bound on the scaling associated with any execution which switches infinitely between the regions. The other possibility is for an execution to eventually enter a region and remain there for ever. Again, condition $G1$, ensures that the scaling associated with the execution is globally bounded. Hence, if $G1$ and $G2$ hold, given any ϵ, by choosing $\delta < \epsilon/b$, where b is this global bound, we can ensure that all executions starting in the δ ball around the origin remain in an ϵ neighborhood of the origin, thereby ensuring Lyapunov stability.

In the case of asymptotic stability, for executions which eventually enter and remain in a region, condition $G3$ ensures that they converge to the origin. Note that since the weight associated with the simple cycles is $< 1 - \gamma$ for some $\gamma > 0$, taking a cycle once takes the execution closer to the origin by a factor of $1 - \gamma$. This observation can be exploited to prove that the executions switching between the regions infinitely often eventually converge to the origin, since they follow the simple cycles infinitely often.

Remark 2. *One of the main highlights of the quantitative abstraction based stability analysis is that the method returns a counter-example in the event of a failure, indicating a potential reason for instability. For instance, a cycle of weight greater than one in the weighted graph expresses the possible existence of an infinite diverging execution along this cycle.*

A method for stability verification of hybrid systems based on graph decomposition and analysis of properties along cycles is presented in [7], however, while [7] is based on Lyapunov functions, our method considers reachability relation computation as the building block.

4.4 Connecting Quantitative Predicate Abstraction and Continuous Simulations

A formal connection between the concrete hybrid system and the abstract weighted graph can be established by interpreting the latter as a one dimensional hybrid system. More precisely, we construct two one dimensional hybrid systems corresponding to a weighted graph $\mathcal{G}$, namely, $\mathcal{H}_\mathcal{G}$ and $\mathcal{H}_\mathcal{G}^{conv}$. $\mathcal{H}_\mathcal{G}$ corresponds to a set of executions σ of a one dimensional system, which can be split into a finite or infinite sequence $\sigma_0\sigma_1\dots$ such that there is a path in the graph $\pi = \pi_0\pi_1\dots$ of the same length and the scaling associated with σ_i is bounded by the weight on the edge π_i. $\mathcal{H}_\mathcal{G}^{conv}$ corresponds to executions of $\mathcal{H}_\mathcal{G}$ which in addition satisfy the "conv" condition on the final region if the path associated with it is finite.

Theorem 4.3. *[12]*

- *$\mathcal{H}_\mathcal{G}$ continuously simulates $\mathcal{H}$;*
- *$\mathcal{H}_\mathcal{G}^{conv}$ continuously simulates $\mathcal{H}$.*

Therefore, from Theorem 4.1, the Lyapunov stability of $\mathcal{H}_\mathcal{G}$ implies the Lyapunov stability of $\mathcal{H}$ and the asymptotic stability of $\mathcal{H}_\mathcal{G}^{conv}$ implies the asymptotic stability of $\mathcal{H}$. Further, conditions $G1$ and $G2$ capture the Lyapunov stability of $\mathcal{H}_\mathcal{G}$, and conditions $G3$ and $G4$ capture the asymptotic stability of $\mathcal{H}_\mathcal{G}^{conv}$.

Remark 3. *More importantly, Theorem 4.3 provides a formal foundation for the quantitative predicate abstraction. As shown in [12], one can formally state that adding new predicates provides an abstraction closer to the original system in the ordering imposed by continuous simulations on the weighted graphs.*

4.5 Computing the Weighted Graph

The main computational challenge with the quantitative predicate abstraction is the computation of the predicate $\mathit{ReachRel}_{R_1,R_2}$ and solving the corresponding optimization problem for computing the weight on the edge. The other challenge is the computation of the label "conv". In this section, we consider two classes of hybrid systems — those in which the invariants and guards are defined by linear constraints, and the dynamics are defined by polyhedral inclusions $\dot{x} \in P$, referred to as polyhedral hybrid systems (*PHA*), or linear dynamics $\dot{x} = Ax$, referred to as linear hybrid systems (*LHA*). We discuss how to compute the QPA for these systems.

4.5.1 Polyhedral Hybrid Systems

A solution of $\dot{x} \in P$ is a trajectory which follows some vector in P. Note that such a solution is finitely varying with respect to a polyhedral partition, since, it never visit a region it has exited. Further, if the execution does not change modes between the regions R_1 and R_2, then the predicate $\mathit{ReachRel}_{R_1,R_2}(s_1, s_2)$ is equivalent to $s_1 \in R_1, s_2 \in R_2, \exists t, \exists v \in P, s_2 = s_1 + vt$. Note that $s_i \in R_i$ is equivalent to s_i satisfying the linear constraints associated with R_i; similarly, the last constraint is equivalent to $(s_2 - s_1)/t$ satisfying the linear constraints associated with P. Hence, the predicate *ReachRel* can be expressed using an existentially quantified first-order logic formula with only conjunctions. In general, the execution from R_1 to R_2 may change modes an unbounded number of times, especially, when the hybrid system graph has a cycle; however, we show that for the purpose of capturing the *ReachRel* it suffices to consider certain "executions" with bounded number of mode switches. Hence, we can express *ReachRel* as an SMT formula over $(\mathbb{R}, <, +)$. For further details, see [9, 10].

In terms of computing the label "conv", one can computationally determine whether there exists an execution which diverges in a particular region R — this happens exactly when there is some vector in the polyhedron P corresponding to some location whose invariant contains R, that belongs to the cone of R at the origin. In fact, we can show that the weighted graph construction exactly characterizes Lyapunov and asymptotic stability in the case of planar systems; hence, we have decidability [13].

4.5.2 Linear Hybrid Systems

For linear dynamics, it is not possible to compute the *ReachRel* exactly. There is a lot of work on constructing over-approximations of *ReachRel*, and bounded error approximations can be obtained provided there is a bound on the time the executions spend in a region. Tools such as SpaceEx [5] can be used for this purpose. However, when there is no bound on the time spent, or in the presence of mode switches, these techniques and tools do not suffice; they do not reach a fixpoint. Hence, we are currently focusing on hybridization techniques for stability analysis, which provide bounds on the scalings rather than bounds on the reachable set in a finite time horizon.

4.6 AVERIST: Algorithmic VERIfier for STability

`AVERIST` [11] is a software tool for stability analysis of hybrid systems. It implements the quantitative predicate abstraction presented in this paper for the class of polyhedral switched systems and the stability analysis based on the weighted graph. It has been implemented in Python. It uses Parma Polyhedra Library (PPL) [4] to manipulate polyhedral sets. The GNU Linear Programming Kit GLPK [1] solver is the linear optimization tool used for computing the weights. The NetworkX Python package [2] is used to define and analyse graphs. Our experiments with polyhedral hybrid systems indicate that the approach is scalable. More importantly, it returns a counter-example which can be examined manually or automatically to refine the abstraction.

5 Conclusions

In this paper, we summarized the algorithmic approach for stability verification that we have developed. There are several interesting future directions. We are currently investigating methods to deal with computational issues associated with extending the results to hybrid systems with linear and non-linear dynamics. We are also exploring compositional techniques for stability analysis.

References

[1] Glpk: https://www.gnu.org/software/glpk/.

[2] NetworkX: https://networkx.github.io/.

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

[4] R. Bagnara, P. M. Hill, and E. Zaffanella. The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems. *Science of Computer Programming*, 72(1–2):3–21, 2008.

[5] Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. Spaceex: Scalable verification of hybrid systems. In Shaz Qadeer Ganesh Gopalakrishnan, editor, *Proc. 23rd International Conference on Computer Aided Verification (CAV)*, LNCS. Springer, 2011.

[6] Thomas A. Henzinger. The Theory of Hybrid Automata. In *Proceedings of the IEEE Symposium on Logic in Computer Science*, pages 278–292, 1996.

[7] Jens Oehlerking. *Decomposition of Stability Proofs for Hybrid Systems*. PhD thesis, Carl von Ossietzky University of Oldenburg, Department of Computer Science, Oldenburg, Germany, 2011.

[8] Pavithra Prabhakar, Geir E. Dullerud, and Mahesh Viswanathan. Pre-orders for reasoning about stability. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pages 197–206, 2012.

[9] Pavithra Prabhakar and Miriam García Soto. Abstraction based model-checking of stability of hybrid systems. In *Proceedings of the International Conference on Computer Aided Verification*, 2013.

[10] Pavithra Prabhakar and Miriam García Soto. An algorithmic approach to stability verification of polyhedral switched systems. In *American Control Conference*, 2014.

[11] Pavithra Prabhakar and Miriam García Soto. Averist: An algorithmic verifier for stability. In *Proceedings of the 8th International Workshop on Numerical Software Verification (NSV)*, 2015. to appear.

[12] Pavithra Prabhakar and Miriam García Soto. Foundations of quantitative predicate abstraction for stability analysis of hybrid systems. In *Proceedings of the International Conference on Verification, Model Checking, and Abstract Interpretation*, pages 318–335, 2015.

[13] Pavithra Prabhakar and Mahesh Viswanathan. On the decidability of stability of hybrid systems. *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, 2013.