

Modular Verification of Interconnected Families of Uniform Linear Hybrid Automata

Matthias Horbach

Viorica Sofronie-Stokkermans

University Koblenz-Landau, Koblenz, Germany
and Max-Planck-Institut für Informatik, Saarbrücken, Germany

Abstract: We provide a mathematical model for unbounded parallel compositions of structurally similar linear hybrid automata, whose topology and flows are described using parametrized formulas. We study how the analysis of safety properties for the overall system can be performed hierarchically, using quantifier elimination to derive conditions on the parameters of individual components.

1 Introduction

We study possibilities of using hierarchical reasoning, quantifier elimination and model generation for the analysis and verification of families of structurally similar parametric hybrid systems. We illustrate our method using a system of water tanks connected in a simple linear topology.

2 Linear Hybrid Automata

A *hybrid automaton* (HA for short) is a tuple $S = (X, Q, \text{flow}, \text{inv}, \text{init}, E, \text{guard}, \text{jump})$. X is a set of variables, Q a set of control modes and E a finite multiset of transitions between modes. For each $q \in Q$, (i) a predicate flow_q over the variables in X and their first derivatives specifies the continuous dynamics, (ii) a predicate inv_q over X defines invariant conditions, and (iii) a predicate init_q over X defines the initial states for control mode q . For each $e \in E$, (i) a predicate guard_e over X restricts states in which the transition can be taken, and (ii) a predicate jump_e over $X \cup X'$ (a copy of X whose elements are “primed”) states how variables are reset during the transition.

A hybrid automaton S is *linear* if it satisfies the following two requirements [3]:

1. *Linearity:* For every control mode $q \in Q$, the flow condition, the invariant condition, and the initial condition are convex linear predicates, i.e. finite conjunctions of strict or non-strict linear inequalities. For every control switch, the guard and reset conditions are convex linear predicates. In addition, as in [1, 2], we assume that the flow conditions are conjunctions of *non-strict* linear inequalities.

2. *Flow independence:* The flow condition of every mode is a predicate over the variables in $\dot{X}$ only (and does not contain any variables from X). This requirement ensures that the possible flows are independent from the values of the variables, and only depend on the control mode.

Parametric Linear Hybrid Automata. We consider *parametric* linear hybrid automata (PLHA) (cf. also [1, 2]), defined as linear hybrid automata for which a set $\Sigma_P = P_c \cup P_f$ of parameters is specified (consisting of parametric constants P_c and parametric functions P_f) with the difference that for every control mode $q \in Q$ and every mode switch e :

- (1) the linear constraints in the invariant conditions inv_q , initial conditions init_q , and guard conditions guard_e are of the form: $g \leq \sum_{i=1}^n a_i x_i \leq f$,
- (2) the inequalities in the flow conditions flow_q are of the form: $\sum_{i=1}^n b_i \dot{x}_i \leq b$,
- (3) the linear constraints in jump_e are of the form $\sum_{i=1}^n b_i x_i + c_i x'_i \leq d$,

where the coefficients a_i, b_i, c_i and the bounds b, d are either numerical constants or parametric constants in P_c ; and g and f are (i) constants or parametric constants in P_c , or (ii) parametric functions in P_f satisfying the convexity (for g) resp. concavity condition (for f), or concrete functions with these convexity/concavity properties such that $\forall t (g(t) \leq f(t))$. The flow independence conditions hold as in the case of linear hybrid automata.

Uniform Parametric Linear Hybrid Automata. The fact that we consider PLHAs allows us to considerably simplify the description of such hybrid automata: We regard then as *uniform parametric hybrid automata* (UPLHA), i.e. systems in which modes have a uniform, but parametric, description. The differences between various modes are expressed not by different shapes of the predicates, but only by different properties of the parameters. For UPLHAs we have the following types of state change:

(Jump) The change of the control location from mode q to mode q' is simply due to an update of the values of the parameters which control the flow. The values of data variables are updated according to the jump conditions.

(Flow) For fixed values of the parameters the state can change due to the evolution in a given control mode over an interval of time: the values of data variables change in a continuous manner according to the flow rules of the current control location for the given values of the parameters.

We have proven that every parametric LHA can be represented as a uniform parametric LHA.

Example 1 Consider a controller that tries to keep the water level in a tank within a safe region, i.e. below a critical level $\text{l}_{\text{overflow}}$, by opening and closing an outlet valve (Figure 1, left). It is described by an LHA with two modes: For the mode in which the valve is closed, the flow is

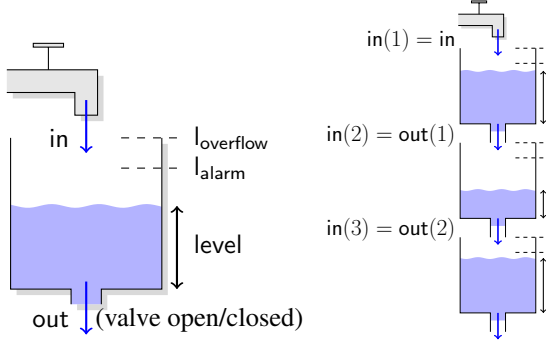


Figure 1: A single water tank and a system of tanks

$\text{level} = \text{in}$, for the mode in which the valve is open we have $\text{level} = \text{in} - \text{out}$. The difference between the two modes is expressed only in the fact that different constants are used in the differential equations which describe the flows in the two modes: the coefficient of out is either 1 or 0.

The difference between the mode in which the valve is open and the mode in which the valve is closed is in the value of the constant in the flow description: $\text{level} = c$. For the mode in which the valve is closed, $c = \text{in}$, for the mode in which the valve is open, $c = \text{in} - \text{out}$. A jump from one mode to the other switches the value of c from in to $\text{in} - \text{out}$ and vice-versa.

3 Systems of Uniform Hybrid Automata

We consider systems of hybrid automata of the form $\{S_i \mid i \in I\}$, where I is an index set (with an underlying structure modeling neighborhood and/or communication) and each S_i is a (uniform) linear hybrid automaton. We assume that all systems S_i have a similar description, using indexed variants of the same continuous variables (cf. [4]). We consider “global” safety properties of the form $\forall i \Psi(i)$. An example of a verification task is to show that the formula is invariant under jumps and flows. We showed that, in this setting, many verification tasks can be decomposed modularly to the verification of a bounded number of components. Moreover, we can synthesize requirements for the parameters that guarantee safety. Because of space limitations, we illustrate the ideas on an example.

Example 2 ([4]) Consider a family of n water tanks with a uniform description (e.g. as in Figure 1, right), each modeled by the hybrid automaton S_i . Assume that every S_i has one continuous variable level_i (representing the water level in S_i), and that the input and output in mode q are described by parameters in_i and out_i . Every S_i has only one mode, in which the water level evolves according to rule $\text{level}_i = \text{in}_i - \text{out}_i$. We write $\text{level}(i, t)$, $\text{in}(i)$, and $\text{out}(i)$ instead of $\text{level}_i(t)$, in_i , and out_i , respectively.

Assume that the water tanks are interconnected in such a way that the input of system S_{i+1} is the output of system S_i . A global constraint describing the communication of the systems is therefore:

$$\forall i (2 \leq i \leq n-1 \rightarrow (\text{in}(i) = \text{out}(i-1)) \wedge \text{in}(1) = \text{in}).$$

An example of a “global” update describing the evolution of the systems S_i during a flow in interval $[t_0, t_1]$:

$$\forall i (\text{level}(i, t_1) = \text{level}(i, t_0) + (\text{in}(i) - \text{out}(i))(t_1 - t_0)).$$

Let $\Psi(t) = \forall i (\text{level}(i, t) \leq l_{\text{overflow}})$ be the safety property of our system that we are interested in. Assume that $\forall i (\text{in}(i) \geq 0 \wedge \text{out}(i) \geq 0)$. We generate a formula which guarantees that Ψ is an invariant: We start with the following formula (for simplicity of presentation we already replaced $\text{in}(i)$ with $\text{out}(i-1)$):

$$\begin{aligned} \exists t_0, t_1 \quad & t_0 < t_1 \wedge \forall i (\text{level}(i, t_0) \leq l_{\text{overflow}}) \\ & \wedge \exists j (\text{level}(j, t_1) > l_{\text{overflow}}) \\ & \wedge \forall i ((i=1 \wedge \text{level}(1, t_1) = \text{level}(1, t_0) + (\text{in} - \text{out}(1))(t_1 - t_0)) \\ & \quad \vee (i > 1 \wedge \text{level}(i, t_1) = \text{level}(i, t_0) \\ & \quad \quad + (\text{out}(i-1) - \text{out}(1))(t_1 - t_0))). \end{aligned}$$

After Skolemization, quantifier elimination and some simplification, we obtain

$$\begin{aligned} \forall i \quad & (i=1 \rightarrow (\text{in} - \text{out}(i_0)) \leq 0) \\ & \wedge (i > 1 \rightarrow (\text{out}(i-1) - \text{out}(i)) \leq 0). \end{aligned}$$

This means that Ψ is an invariant for the family of systems if, and only if, this condition is satisfied.

4 Conclusion

We presented a way of representing systems of similar hybrid automata which allows us to reduce the problem of checking invariance of safety conditions under jumps and flows to checking satisfiability of ground formulae w.r.t. background theories. We illustrated the types of problems which can be solved for a system $\{S_i \mid i \in I\}$ of parametric linear hybrid automata, where I is a list (here modeled by the set of natural numbers). Our method can also be applied to more complex topologies (e.g. systems of water tanks where I has a tree structure). Similar results can be obtained if updates are caused by changes in the topology (insertion or deletion of water tanks in the system in our example). In the future, we want to make our approach fully automatic and analyze its complexity.

Acknowledgements: This work was partly supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center “Automatic Verification and Analysis of Complex Systems” (SFB/TR 14 AVACS). See www.avacs.org for more information.

References

- [1] W. Damm, C. Ihlemann, V. Sofronie-Stokkermans. Decidability and complexity for the verification of reasonable linear hybrid automata. *Proc. HSCC 2011*, 73–82, ACM, 2011.
- [2] W. Damm, C. Ihlemann, V. Sofronie-Stokkermans. PTIME Parametric Verification of Safety Properties for Reasonable Linear Hybrid Automata. *Mathematics in Computer Science* 5(4): 469–497, 2011.
- [3] T.A. Henzinger, P.W. Kopke, A. Puri, P. Varaiya. What’s decidable about hybrid automata? *Journal of Computer and System Sciences* 57(1): 94–124, 1998.
- [4] V. Sofronie-Stokkermans. Hierarchical Reasoning and Model Generation for the Verification of Parametric Hybrid Systems. *Proc. CADE 2013*, 360–376, Springer, 2013.