Accounting for communication delays in hybrid systems

Guruprerana Shabadi

October 28, 2021

1 Notation

Definition 1 (Lazy hybrid automaton in the continuous semantic)**.** Consider a vector of system variables $\mathbf{x} = (x_1, \ldots, x_n) \in X$, a vector of *input system variables* $\mathbf{y} = (y_1, \ldots, y_n) \in Y$, control inputs $\mathbf{u} \in U$ and disturbances $\mathbf{w} \in W$. Then, a lazy hybrid automaton (LHA) is a tuple $\mathscr{H} : (L, E, I, F, G, B, R)$ consisting of:

1. A finite set of *locations* $Q = \{q_1, \ldots, q_k\}$ and *transitions* that form edges between states $E \subseteq Q \times Q$.
2. A map I that associates with each state $q \in Q$ an *invariant* $I(q)$ which is a predicate over the variables in $(\mathbf{x}, \mathbf{y})$.
3. A map F that associates each mode $q \in Q$ with a *vector field* $F_q : X \times U \times W \to (\mathscr{T}X)$ that forms the RHS of the ODE $\dot{x}(t) = F_q(\mathbf{x}, \mathbf{u}, \mathbf{w})$. F_q is assumed to be Lipschitz continous over $\mathbf{x}$ and continous over the remaining variables.
4. A *guard map* G that associates a *guard* condition $G(q_i, q_j)$ with each transition, which is also a predicate over $(\mathbf{x}, \mathbf{y})$ like the invariant.
5. A map of *time bounds* which associates each system variable in $(\mathbf{x}, \mathbf{y})$ to a pair of non-negative reals (l_{x_i}, u_{x_i}) or (l_{y_i}, u_{y_i}) respectively. These bounds define the time interval that we are allowed to look backward with respect to each system variable in $(\mathbf{x}, \mathbf{y})$ to evaluate its value.
6. A *reset map* R that associates each transition with an *update function* $R(q_i, q_j) : X \to X$.

The set of initial conditions for a LHA is given by a map for the initial states of the variables $(\mathbf{x}_0, \mathbf{y}_0) : [-\max_{x \in \mathbf{x}, y \in \mathbf{y}}(l_x, l_y), 0] \to X \times Y$, an initial location $q_0 \in Q$, the control inputs $\mathbf{u} : [0, T] \to U$, and the disturbance inputs $\mathbf{w} : [0, T] \to W$, where $T > 0$ is a time horizon for an execution of the LHA.

Note that the vector of *input* variables are system variables which evolve in other systems and appear to us as inputs. Also, we introduce delays in the observation of the system variables to represent communication delays in the observation of these variables.

Corresponding to the delays, we also introduce **new semantics for guard and invariant evaluation** along with this new definition of a lazy hybrid automaton (LHA) which extends the definition of a standard hybrid automaton. Before introducing these new semantics, we define the *timings*

map. Given a LHA $\mathscr{H}$, and for a given time $t \geq 0$, we can define a *timings map* τ_t which associates every system variable in $\{x_1, \ldots, x_n, y_1, \ldots, y_m\}$ with a time in the interval $\tau(s) \in [t - l_s, t - u_s]$ where s is one of the system variables.

We are now ready to define the new semantics of the guard and invariant evaluation. For the guard evaluation, we impose that a transition can take place from a location q to q' at time $t \in [0, T]$, if $(q, q') \in E$ and there exists a timings map τ_t at time t, such that

$$(x_1(\tau_t(x_1)), \ldots, x_n(\tau_t(x_n)), y_1(\tau_t(y_1)), \ldots, y_m(\tau_t(y_m))) \models G(q, q')$$

For the invariants too, we have a similar semantic. We say that the invariants hold during an execution of the system if, for all $t \in [0, T]$, there exists a timings map τ_t at time t, such that

$$(x_1(\tau_t(x_1)), \ldots, x_n(\tau_t(x_n)), y_1(\tau_t(y_1)), \ldots, y_m(\tau_t(y_m))) \models I(q)$$

where q is the location of the system at the time $t \in [0, T]$.

1.1 Execution traces in a lazy hybrid automaton

Let $\mathscr{H} : (L, E, I, F, G, B, R)$ be a lazy hybrid automaton and let $[-\max_{x \in \mathbf{x}, y \in \mathbf{y}}(l_x, l_y), T \geq 0$ be a time horizon over which we can have an execution of this lazy hybrid automaton. Further, let $\mathbf{u} : [0, T] \to U$ be a control signal input and $\mathbf{w} : [0, T] \to W$ be a disturbance input. The rest of the initial conditions of the LHA are given by discrete state of the automaton $q_i \in Q$ and an initial state of the variables $(\mathbf{x}_0, \mathbf{y}_0) : [-\max_{x \in \mathbf{x}, y \in \mathbf{y}}(l_x, l_y), 0] \to X \times Y$. We can then define the semantic of an execution trace on a LHA as a sequence of jumps and flows (similar to how they do it in [1]):

- A flow $(q, \mathbf{x}, t) \rightsquigarrow (q, \mathbf{x}, t + \delta)$ for some $\delta \geq 0$ is a solution to the ODE $\dot{\mathbf{x}} = F_q(\mathbf{x}, \mathbf{u}, \mathbf{w})$ starting from the initial condition $t_0 = t$ and $\mathbf{x}(t)$. Since F_q is Lipschitz continous over $\mathbf{x}$, the trajectory is uniquely defined. Of course, the invariant is also expected to hold throughout the flow period, i.e., for all $\lambda \in [t, t + \delta]$, there exists a timings map τ_λ at time λ, such that

  $$(x_1(\tau_t(x_1)), \ldots, x_n(\tau_t(x_n)), y_1(\tau_t(y_1)), \ldots, y_m(\tau_t(y_m))) \models I(q)$$

- A jump

  $$(q, \mathbf{x}, t) \xrightarrow{\tau_t} (q', \mathbf{x}', t)$$

  is an instantaneous transition from the discrete state q to q', where $(q, q') \in E$. the timings map τ_t satisfies

  $$(x_1(\tau_t(x_1)), \ldots, x_n(\tau_t(x_n)), y_1(\tau_t(y_1)), \ldots, y_m(\tau_t(y_m))) \models G(q, q')$$

  where $G(q, q')$ is the guard set of the transition. Lastly, we obtain $\mathbf{x}' = R_{(q,q')}(\mathbf{x})$, by applying the reset map of the transition.

A sequence of these jumps and flows over a time horizon $T > 0$ defines an execution trace of the given LHA. For a given time horizon $T > 0$, and for a given set of initial conditions of the LHA, we can define the set of execution traces accepted by a LHA $\mathscr{H}$ and we denote this set by $\Gamma_{\mathscr{H}}$.

1.2 Parallel composition of lazy hybrid automata

Given two lazy hybrid automata which might share input variables with each other we would like to be able to define the parallel composition of these two automata resulting in one single LHA. *Sharing input variables* amounts to the fact that there might be some system variables which evolve in one LHA that might appear as inputs in the other LHA.

In order to formally define this parallel composition, consider two LHA given by

- A vector of system variables for the first LHA $\mathbf{x} = (x_1, \ldots, x_k, s_1, \ldots, s_p)$, where $(s_1, \ldots, s_p)$ is the vector of system variables evolving in the first LHA which appear as inputs in the second LHA.

  The input variables for this LHA are $\mathbf{y} = (y_1, \ldots, y_m, s'_1, \ldots, s'_{p'})$. Again, $(s'_1, \ldots, s'_{p'})$ is the vector of system variables of the second LHA which appear as input variables in this first LHA.

  Lastly, the tuple describing the first LHA is $\mathscr{H} : (L, E, I, F, G, B, R)$.

- Similarly, the vector of system variables for the second LHA would be $\mathbf{x}' = (x'_1, \ldots, x'_{k'}, s'_1, \ldots, s'_{p'})$, and the vector of input variables, $\mathbf{y}' = (y'_1, \ldots, y'_{m'}, s_1, \ldots, s_p)$.

  The tuple is $\mathscr{H}' : (L', E', I', F', G', B', R')$.

The first step towards defining the parallel composition is to introduce copy variables for $(s_1, \ldots, s_p)$ and $s'_1, \ldots, s'_{p'}$ which we denote by $(c_1, \ldots, c_p)$ and $(c'_1, \ldots, c'_{p'})$ respectively. When we say *copy* variables, we mean that the new variables have the exact same dynamics as the original variables. Now, we can define the parallel composition of $\mathscr{H}||\mathscr{H}'$ as the LHA defined over the system variables

$$\mathbf{x}_c = (x_1, \ldots, x_k, s_1, \ldots, s_p, c_1, \ldots, c_p, x'_1, \ldots, x'_{k'}, s'_1, \ldots, s'_{p'}, c'_1, \ldots, c'_{p'})$$

along with the tuple of the parallel composition $\mathscr{H}_c : (L_c, E_c, I_c, F_c, G_c, B_c, R_c)$ where

- The new vector input variables is given by

  $$\mathbf{y}_c = (y_1, \ldots, y_m, c'_1, \ldots, c'_{p'}, y'_1, \ldots, y'_{m'}, c_1, \ldots, c_p)$$

  Here, we remove the shared system variables which appeared previously as inputs. However, they are no longer inputs when we compose the two automata since they evolve in the composed LHA.

- The new set of locations is just $L_c = L \times L'$ and there exists an edge from $(q_a, q'_a) \in L_c$ to $(q_b, q'_b) \in L_c$ in E_c if there is an edge $(q_a, q_b) \in E$ and $(q'_a, q'_b) \in E'$.

- For each location $(q, q') \in L_c$, we associate the invariant $I_c((q, q')) = I(q) \wedge I'(q')$. However, within $I(q)$, since there might be variables of the form $(s'_1, \ldots, s'_{p'})$, we replace these by their copies $(c'_1, \ldots, c'_{p'})$. Within $I'(q')$ we similarly replace all the variables of the form $(s_1, \ldots, s_p)$ with their copies $(c_1, \ldots, c_p)$. Note that these are the system variables of the LHA which appear as input variables to the other LHA. However, observe that we might still have variables of the form $(s_1, \ldots, s_p)$ in $I(q)$, but we do not replace these with their copies

because these are system variables within $\mathscr{H}_1$ and there are no delays associated with these within $\mathscr{H}_1$.

- For the vector field at each location (q, q'), we have $\dot{\mathbf{x}}_c = (\dot{\mathbf{x}}, \dot{\mathbf{x}}') = (F(\mathbf{x}, \mathbf{u}, \mathbf{w}), F'(\mathbf{x}', \mathbf{u}', \mathbf{w}'))$. The only change required would be to ensure to make appropriate changes within the dynamics to ensure that the copies of the variables obtain the same dynamics as their counterparts, i.e., $(c_1, \ldots, c_p)$ all share the same differential equation as $s_1, \ldots, s_p$ and similarly with $(c'_1, \ldots, c'_{p'})$.
- The new guard conditions associated to each transition $((q_a, q'_a), (q_b, q'_b)) \in E_c$ is given by $G_c((q_a, q'_a), (q_b, q'_b)) = G(q_a, q_b) \wedge G'(q'_a, q'_b)$. However, similar to the invariants, we need to replace the variables $(s'_1, \ldots, s'_{p'})$ appearing in $G(q_a, q_b)$ with their copies and similarly with $G'(q'_a, q'_b)$.
- With respect to the new time bounds B_c, we need to have time bounds associated to each of the system and input variables.
  - For the variables $(x_1, \ldots, x_k, s_1, \ldots, s_p)$ and $(y_1, \ldots, y_m)$ we retain the same time bounds from B in $\mathscr{H}$.
  - For the variables $(c'_1, \ldots, c'_{p'})$, we associate the time bounds associated to $(s'_1, \ldots, s'_{p'})$ in B since these are input variables in $\mathscr{H}$.
  - For the variables $(x'_1, \ldots, x'_{k'}, s'_1, \ldots, s'_{p'})$ and $(y'_1, \ldots, y'_{m'})$ we retain the same time bounds associated with these variables in B'.
  - Lastly, for the variables $(c_1, \ldots, c_p)$, we associate the time bounds associated with $(s_1, \ldots, s_p)$ in B' since these are input variables in $\mathscr{H}'$.
- Finally, for the reset maps, we would have for each transition $((q_a, q'_a), (q_b, q'_b)) \in E_c$, the associated reset map defined by

$$R_c((q_a, q'_a), (q_b, q'_b))(\mathbf{x}, \mathbf{x}') = (R(q_a, q_b)(\mathbf{x}), R'(q'_a, q'_b)(\mathbf{x}'))$$

  However, we should also keep in mind to modify this reset function appropriately so as to also reset the copied variables to the same values as their originals.

The main reason why we need to introduce copy variables of our system variables within this parallel composition is because the shared system variables between the two LHA might have different time bounds (delays) associated with each of them. We need to retain both these different delays on the same system variables within the invariants and the guard conditions.

2 Reduction of lazy hybrid automata

In this section we explore some subclasses of lazy hybrid automata and show how they can be translated into regular hybrid automata.

2.1 Translation of lazy hybrid automata without invariants

Consider a subclass of LHA which we call LHA*. $\mathscr{H}^* : (L, E, I, F, G, B, R)$ is a LHA* if it satisfies the following conditions:

- *Invariants*: The invariants of $\mathscr{H}^*$ at every location given by $I(q)$ is **only** a predicate over the variables in $\mathbf{x}$ and does not contain any variables from the input variables $\mathbf{y}$.
- *Strongly non-zeno*: There exists an upper bound $K \in \mathbb{N}$ on the number of transitions that can be taken in any given time period of length $\max_{i\in\{1,\ldots,m\}}(u_i - l_i)$ where $(l_i, u_i) \in B$ are the time bounds associated to the input variables.

We now have the following result with this restricted subclass LHA*

Theorem 1 (Translation from LHA* to HA)**.** *For every LHA* $\mathscr{H} : (L, E, I, F, G, B, R)$, there exists a regular hybrid automaton $\mathscr{R}$ such that they both accept the same set of execution traces, i.e.,* $\mathbf{T}_{\mathscr{H}} = \mathbf{T}_{\mathscr{R}}$.

Proof. Firstly, we construct the translated hybrid automaton and then proceed to show that their execution traces do indeed coincide. □

References

[1] Deshmukh, J. V. and Sankaranarayanan, S. 2019. *Formal Techniques for Verification and Testing of Cyber-Physical Systems.*