Ensuring Sound Numerical Simulation of Hybrid Automata

Yerang Hur
Department of Computer and Information Science, University of Pennsylvania, USA
yehur@posdata-usa.com

Jae-Hwan Sim
Department of Computer Science and Engineering, Korea University, Korea
jhsim@formal.korea.ac.kr

Jesung Kim
Department of Computer and Information Science, University of Pennsylvania, USA
jesung.kim@mathworks.com

Jin-Young Choi†
Department of Computer Science and Engineering, Korea University, Korea
choi@formal.korea.ac.kr

Received 24 November 2008; Accepted 24 June 2009

A hybrid system is a dynamical system in which states can be changed continuously and discretely. Simulation based on numerical methods is the widely used technique for analyzing complicated hybrid systems. Numerical simulation of hybrid systems, however, is subject to two types of numerical errors: truncation error and round-off error. The effect of such errors can make an impossible transition step to become possible during simulation, and thus, to generate a simulation behavior that is not allowed by the model. The possibility of an incorrect simulation behavior reduces con.dence in simulation-based analysis since it is impossible to know whether a particular simulation trace is allowed by the model or not. To address this problem, we define the notion of Instrumented Hybrid Automata (IHA), which considers the effect of accumulated numerical errors on discrete transition steps. We then show how to convert Hybrid Automata (HA) to IHA and prove that every simulation behavior of IHA preserves the discrete transition steps of some behavior in HA; that is, simulation of IHA is sound with respect to HA.

---

†: corresponding author

Categories and Subject Descriptors: Real-Time Systems [**Systems & Architecture**]

General Terms: Hybrid Systems, Hybrid Automata

Additional Key Words and Phrases: Numerical Simulation, Numerical Errors, Alternating Runs, Instrumented Hybrid Automata

1. INTRODUCTION

Hybrid systems are extended finite state machines with continuous dynamics. There are many embedded system applications that can be modeled as hybrid systems, such as automotive systems, avionic systems, coordinated robot systems, and medical devices. For example, in a coordinated robot navigation system, the system controls the collection of robots using one set of control laws until an obstacle is detected. When a robot detects an obstacle, a different set of control laws is used to avoid the obstacle [Fierro et al. 2002].

When developing a complex embedded system, it is common to model it as a hybrid system and analyze it before implementation. The most commonly used analysis technique for hybrid systems is to simulate using numerical methods [Chutinan and Krogh 2003; Hickey and Wittenberg 2004; Henzinger et al. 2000]. During simulation, one is concerned with whether or not good or bad states are reachable from an initial state. However, it is well known that numerical errors, such as truncation and round-off errors, due to numerical computation with a finite precision can cause impossible transitions of the model to occur during simulation [Gear 1971; Jain 1979; Press et al. 1999; Abate et al. 2006; Donzé and Maler 2007].

This paper presents a framework in which such erroneous transitions can be prevented during simulation. Our framework, called Instrumented Hybrid Automata (IHA), guarantees that all discrete transitions taken are exactly those allowed in the original model. Contrast to the approach described in this paper, other researchers have developed related, but different, techniques. In HyTech+ [Henzinger et al. 2000] and CheckMate [Chutinan and Krogh 2003], techniques use interval numerical method and overapproximation respectively. Although interval method can avoid round-off error and guarantee true solution within validated bounds, such bound may be unacceptably wide in the worst case, and overapproximation cannot avoid numerical error completely. Also, they have restriction. For example, HyTech does not allow ODEs (Ordinary Differential Equations) which can specify continuous physical systems, and CheckMate can only use restricted Hybrid Automata which is subclass of Hybrid Automata. Another techniques research aspect of correctness by focusing on how to avoid missing events that trigger discrete transitions during simulation [Esposito et al. 2001; Park and Barton 1996; Abate et al. 2006], but they are different from our approach.

The rest of the paper is organized as follows: We start with defining a variation of hybrid automata in Section 2. Our definition is similar to [Chutinan 1999; Lafferriere et al. 1999] among the class of hybrid automata [Alur et al. 1995; Chutinan 1999; Lafferriere et al. 1999; Lynch et al. 1995; Henzinger 1996]. We define the notion of *alternating run* of hybrid automata. Section 3 describes the effect of numerical errors and proposes a formal way of instrumenting the original model to limit the effect of

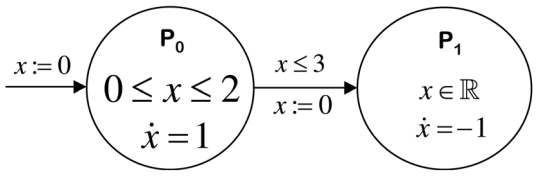


Figure 1. Hybrid Automaton Boo.

numerical errors. We consider only truncation errors in this paper. Then, we prove that every alternating run of an instrumented hybrid automaton is always safe with respect to the original hybrid automaton. Section 4 concludes the paper and discusses future work.

2. HYBRID AUTOMATA

This section defines a hybrid automaton as an extension of the timed automaton [Alur and Dill 1994] to include continuous dynamics in addition to discrete transitions. Among many notions of hybrid automaton [Alur et al. 1995; Chutinan 1999; Lafferriere et al. 1999; Lynch et al. 1995], our definition is most close to the one given in [Chutinan 1999].

Definition 1. (HA). *A hybrid automaton, HA, is a tuple* $A = (P, VC, p_0, F, E, I, G, R, INIT)$, *where*

- P *is a finite set of distinct positions,*
- VC *is a finite set of continuous real variables, where* $|VC| = n$,
- $p_0 \in P$ *is the initial position,*
- $F: P \to \mathcal{F}$ *assigns to* $p \in P$ *an n-dimensional vector field* $F(p) \in \mathcal{F}: \mathbb{R}^n \to \mathbb{R}^n$, *which defines ordinary differential equations satisfying the assumptions for existence and uniqueness of solutions for all variables in VC,*
- $E \subseteq P \times P$ *is a finite set of discrete transitions,*
- $I: P \to (2^{\mathbb{R}})^n$ *assigns the invariant interval to* $p \in P$ *such that* $I(p) \in (2^{\mathbb{R}})^n$ *and, for all* $x \in VC$, *we denote the invariant interval of x at the position p by* $I_x(p)$,
- $G: E \to (2^{\mathbb{R}})^n$ *assigns to* $(p_1, p_2) \in E$ *the guard interval such that for all* $x \in VC$, $G_x((p_1, p_2)) \cap I_x(p_1) \neq \emptyset$, *where* $G_x((p_1, p_2))$ *denotes the guard interval of x,*
- $R: E \times VC \to \mathbb{R}$ *assigns a reset value* $R((p_1, p_2), x) \in I_x(p_2)$ *to a pair* $(p_1,p_2) \in E$ *and* $x \in VC$, *and*
- $INIT: VC \to \mathbb{R}$ *assigns to a variable the initial value satisfying* $INIT(x) \in I_x(p_0)$, *for all* $x \in VC$.

In the rest of the paper, we denote P of A by P_A. Likewise, we use VC_A, p_0^A, F_A, E_A, I_A, G_A, R_A, and $INIT_A$ to denote V_C, p_0, F, E, I, G, R, and $INIT$ of A, respectively. For all $x \in VC_A$, the invariant and the guard intervals of x are denoted by $I_{A,x}$ and $G_{A,x}$, respectively. When it is clear, we omit A. Figure 1 is a simple example of HA called *Boo* with two positions p_0 and p_1 and one continuous variable x of which dynamics at p_0 and p_1 are $\dot{x} = 1$ and $\dot{x} = -1$, respectively. The discrete transition E_{Boo} is $\{(p_0, p_1)\}$, the invariant intervals $I_{Boo,x}(p_0) = [0, 2]$ and $I_{Boo,x}(p_1) = (-\infty, \infty)$, the guard interval

$G_{Boo,x}((p_0, p_1)) = (-\infty, 3]$, the reset value $R_{Boo}((p_0, p_1),x) = 0$, and the initial value of x, $INIT_{Boo}(x) = 0$.

Definition 2. (State of an HA). *Given an HA A, a (time-stamped) state* $s = (p, u, t)$ *is an element of* $P_A \times \mathbb{R}^n \times \mathbb{R}$ *satisfying the following condition: at time t, for all* $x \in VC$, $u(x) \in I_x(p)$, *where* $u(x)$ *is the valuation of x.*

A state (p, u, t) means that at time t the system is at the position p with the valuation u. When a state $s_i = (p_i, u_i, t_i)$ is given, we use $s_i|_p, s_i|_u, s_i|_t$ to denote p_i,u_i,t_i, respectively. In addition, we use $u \in I_A(p)$ if $u(x) \in I_{A,x}(p)$ for all $x \in VC$, and $u \in G_A((p_1, p_2))$ if $u(x) \in G_{A,x}((p_1, p_2))$ for all $x \in VC$.

Definition 3. (Discrete transition step of an HA). *Given an HA A, a pair of states* (s_i,s_j) *is called a discrete transition step if the following conditions are satisfied:*

- $s_i|_t = s_j|_t$,
- $(s_i|_p, s_j|_p) \in E_A$,
- $s_i|_u \in I_A(s_i|_p)$,
- $s_i|_u \in G_A((s_i|_p, s_j|_p))$, *and*
- $s_j|_u(x) = R_A((s_i|_p, s_j|_p),x)$, *for all* $x \in VC$.

A discrete transition is of the form (p_m, p_n), i.e., a directed edge from the node p_m to the node p_n, whereas a discrete transition step is (s_i,s_j) defined in Deffnition 3. We say that a discrete transition $e = (p_1,p_2) \in E_A$ becomes *enabled* in p_1, if $G_A((p_1, p_2))$ is true, and e is *legitimate* if its reset action satisfies the condition $R_A((p_1, p_2), x) \in I_{A,x}(p_2)$, for all $x \in VC_A$. A discrete transition must be both *enabled* and *legitimate* for it to be taken.

Definition 4. (Continuous transition step of an HA). *Given an HA A, a pair of states*

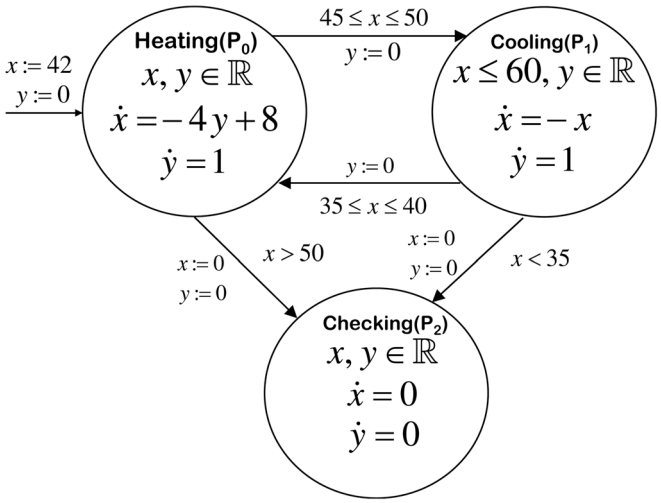


Figure 2. Thermostat: Hybrid Automaton Therm.

(s_i, s_j) is called a continuous transition step if the following conditions are satisfied:

- $s_i|_t < s_j|_t$,
- $s_i|_p = s_j|_p$, *and*
- $s_j|_u$ *is the solution of the initial value problem of the ordinary differential equation, where continuous dynamics is given by the vector field* $F(s_i|_p)$ *and the initial value is* $s_i|_u$ at time $s_i|_t$.

The continuous transition step corresponds to the continuous flow at the position p with the dynamics specified by $F(p)$ from time t_i to time t_j. We use a term *transition step* to refer to both a discrete transition step and a continuous transition step.

Definition 5. (Run of an HA). *A run of an HA A is a finite or an infinite sequence of states, where the following conditions are satisfied:*

- $s_0 = (p_0, INIT_A, 0)$,
- $\forall i,\ s_i|_t \leq s_{i+1}|_t$,
- *.if* $s_i|_t = s_{i+1}|_t$, (s_i, s_{i+1}) *is a discrete transition step at time* $s_i|_t$, and
- *.if* $s_i|_t < s_{i+1}|_t$, (s_i, s_{i+1}) *is a continuous transition step at time* $s_i|_t$.

Figure 2 shows an example of an HA called *Therm* which is thermostat for regulating temperature between 35 to 50 degrees. We will denote heating position by p_0, cooling position by p_1, and checking position by p_2. *Therm* start at the position p_0 and taking a discrete transition (p_0, p_1) at some time while the guard is true. This is a pedagogical example to show the effect of numerical errors. In *Therm*, the transition (p_0, p_2) cannot occur, since the guard $G(p_0, p_2)$ is always false. *Therm* consists of two continuous variables and three positions with dynamics of each position described by Equations (1), (2), and (3):

$$\text{position } p_0 : \dot{x} = -4y + 8,\ \dot{y} = 1,\ x(0) = 42,\ y(0) = 0 \tag{1}$$

$$\text{position } p_1 : \dot{x} = -x,\ \dot{y} = 1, \tag{2}$$

$$\text{position } p_2 : \dot{x} = 0,\ \dot{y} = 0. \tag{3}$$

HA_{TS} starts at the initial position p_0 with dynamics of Equation (1) and the initial values $x(0) = 42$ and $y(0) = 0$. If we solve Equation (1), we get $x(t) = -2t^2 + 8t + 42$, $y(t) = t$. During a run of *Therm*, the transition from a position p_0 may be taken at any time if the following conditions satisfied: the guard of an edge is true, its reset assigns legitimate values, and the invariant of p_0 is not violated. Note that the transition (p_0, p_2) is not supposed to occur in the HA *Therm*, because the maximum value of x at the position p_0 is 50.

Definition 6. (Model of an HA). *For a given HA A, the set of all runs of A is called the model of A.*

Definition 7. (Alternating run of an HA). *A run* $\langle s_0, s_1, ..., s_i, s_{i+1} ....\rangle$ *of an HA A is called an* alternating run *if it satisfies the following conditions: for all* $i \geq 0$,

- *if* (s_i, s_{i+1}) *is a discrete transition step, then* (s_{i+1}, s_{i+2}) *is a continuous transition*

step.

– *if* (s_i, s_{i+1}) *is a continuous transition step, then* (s_{i+1}, s_{i+2}) *is a discrete transition step.*

3. INSTRUMENTED HYBRID AUTOMATA

In this section, we briefly overview numerical methods and identify the possible sources of numerical errors. Then, we explain how the numerical errors can cause an unexpected discrete transition to occur during simulation. To formalize the effect of numerical errors during simulation of an HA, we propose the formalism called Instrumented Hybrid Automata (IHA). We show that the alternating run of the IHA trace is always included in that of the HA trace. Thus, the IHA provides only correct simulation results with regard to alternating runs of the original model.

3.1 Numerical Methods

To make the presentation concrete, we assume that the numerical methods used for initial value problems of ODEs are Runge-Kutta methods, which include Euler's method[1]. Given initial values and a start time, Runge-Kutta (RK) methods compute a solution over an interval of time period by combining intermediate values from Euler-style steps based on a Taylor series expansion up to some degree of orders.

To identify the possible sources of numerical errors, we briefly explain Euler's method and the classical fourth-order RK method (for details see [Gear 1971; Press et al. 1999]). Euler's method is the simplest method computing the approximate solution of differential equation of the form:

$$\dot{x} = f(t, x)\,. \tag{4}$$

Assuming $f(t_0, x_0)$ will remain as a constant from t_0 to t_1, the value of x_1 at t_1 is computed using the following equation:

$$x_1 - x_0 = f(t_0, x_0)(t_1 - t_0). \tag{5}$$

Thus, $x_1 = x_0 + hf(t_0, x_0)$, where h is the integration stepsize. In general, Euler's method computes:

$$x_{i+1} = x_i + hf(t_i, x_i), \text{ where } i = 0, 1, 2, \cdots. \tag{6}$$

Most scientific applications, however, require a higher accuracy than Euler's method provides. A common technique is to use a fourth-order RK method, which combines the four intermediate evaluations of the function f. The fourth-order RK method evaluates $f(t, x)$ at different values of t and x four times and combine the intermediate results to get an approximation of x at time t_{i+1}.

3.2 Numerical Errors

The numerical errors of RK methods are of the two types: the *truncation error* due to truncations in Taylor series expansions and the *round-off error* due to a finite precision of real numbers in the computer. Both the truncation error and the round-off error are accumulated from the first integration to the final integration. In the rest

[1]Euler's method is the first-order Runge-Kutta method.

of the paper, we call the accumulated truncation error *global truncation error* to distinguish from the *local truncation error* referring to the truncation error in a single integration step. Unless the size of h becomes very small, local and global truncation errors are the sources of dominant numerical errors. Note that truncation errors exist even if an infinite precision arithmetic is used since they originate from the truncation of the infinite Taylor expansion.

Euler's method or Taylor series expanded up to the second term is represented as $x(t+h) = x(t)+h\dot{x}(t)+O(h^2)$, where $x(t+h) = x_{i+1}$, $x(t) = x_i$, and $\dot{x}(t) = f(t_i, x_i)$. Therefore, the local truncation error becomes $O(h^2)$; that is, the local truncation error of Euler's method is the second order of h. In general, a k-th order RK method is derived from the Taylor series expansion of the form

$$x_{i+1} = x_i + hf(t_i, x_i) + \frac{1}{2!}h^2\left(\frac{df(t_i,x_i)}{dt}\right) + \cdots + \frac{1}{k!}h^k\left(\frac{df^{k-1}(t_i,x_i)}{dt^{k-1}}\right) + O(h^{k+1}), \text{ where } k > 1. \quad (7)$$

In Equation (6), the local truncation error is $O(h^{k+1})$.

We use the example presented in Figure 2 to show why we need to instrument a given HA to ensure that the numerical simulation follows transitions possible in the HA.

Table I shows how the global error affects the simulation of HA *Therm*. In the table, h is 0.001, x denotes the exact value and x_{Euler} represents the value computed by Euler's method. At all positions of *Therm*, we used the same numerical method and stepsize. The relative global error is denoted by E_g.

In any run of HA *Therm*, the transition from p_0 to p_2 cannot happen since the guard of $G_{Therm}((p_0, p_2))$ is $x > 50$. The maximum value of x at the position p_0 is 50. However, even with a small global error of 0.0008%, the guard becomes true and thus, the simulator can show the run reaching the position p_2 at any time [1.956, 2.045]. Also, Table II shows simulation result which uses RK method in order to acquire high accuracy. In case of RK, similar to Euler's method, the simulator can show the run

Table I. The Affect of Numerical Errors of Euler's Method.

t	x	x_{Euler}	$E_g : (x_{Euler} - x)/x$
1.955	49.995950	49.999860	0.008%
1.956	49.996128	50.000040	0.008%
1.957	49.996302	50.000216	0.008%
…	…	…	…
2.000	50.000000	50.004000	0.008%
…	…	…	…
2.044	49.996128	50.000216	0.008%
2.045	49.995950	50.000040	0.008%
2.046	49.995768	49.999860	0.008%

Table II. The Affect of Numerical Errors of RK Classic Method

t	x	x_{Euler}	$E_g : (x_{RK} - x)/x$
1.998	49.999992	49.999992	0%
1.999	49.999996	50.000000	0.000008%
2.000	50.000000	50.000004	0.000008%
2.001	49.999996	50.000000	0.000008%
2.002	49.999992	49.999992	0%

reaching the position p_2 at $t = 2.000$. In addition, we make sure of same simulation result in Simulink ODE1 and ODE4 solver. Therefore, Incorrect simulation can be generate in most case which use numerical methods.

For these reason, we need to provide a safe way of simulating HA using the notion of Instrumented Hybrid Automaton (IHA).

3.3 The Bound of the Local Truncation Error

Let $N(p)$ and $h(p)$ denote the numerical method and the integration stepsize at the position p, respectively. Also, let $\delta_x(N(p), h(p), T_f, F(p), VC, INIT)$ denote the bound of the maximum local truncation error when x is integrated using a numerical program $N(p)$ with a stepsize $h(p)$ from time 0 to T_f, where $x \in VC$ and its dynamics is defined by $F(p)(x)$.

If $N(p)$ is a *k-th order RK method*, then $\delta_x(N(p), h(p), T_f, F(p), VC, INIT)$ can be computed using the $(k+2)$-*th* term of the Taylor expansion of $x(t + h(p))$ at $x(t)$. To see it, if $N(p)$ is Euler's method, use the Taylor expansion of $x(t + h(p)) = x(t) + h(p) f(t, x) + \frac{1}{2!} h^2(p) f'(\hat{t}, x)$, for some $t < \hat{t} < t + h(p)$. Then, the local truncation error at time $t + h(p)$ is $|x(t + h(p)) - x(t) - h(p) f(t, x)|$. If there exists K such that $|x(t + h(p)) - x(t) - h(p) f(t, x)| \le Kh^2(p)$ for all $0 \le t \le T_f$, δ_x is $Kh^2(p)$.

3.4 The Bound of the Global Truncation Error

Let $e_n(x)$ and $E_n(x)$ denote the local truncation error and the global trunca tion error of x in n-th integration step, respectively. Then, $|E_{n+1}(x)|$ is bounded in $|E_n(x)| + L \cdot h \cdot |E_n(x)| + |e_{n+1}(x)|$, where L is the Lipschitz constant and h is the fixed integration stepsize. Namely, $|E_n(x)| \le ((1 + L \cdot h(p))^n - 1) \cdot (\textit{the bound of local truncation error of } x)/(L \cdot h(p))$. As the maximum local trun cation during numerical integration of x from time 0 to T_f using the numerical program $N(p)$ and the stepsize $h(p)$, is bounded in $\delta_x(N(p), h(p), T_f, F(p), VC, INIT)$, we get the following inequality.

$$|E_n(x)| \le ((1 + L \cdot h(p))^n - 1) \cdot \delta_x/(L \cdot h(p)). \tag{8}$$

We know $1+(Lh) \le 1+(Lh) + \frac{1}{2!}(Lh)^2 + \frac{1}{3!}(Lh)^3 + \cdots = e^{(Lh)}$. Thus, $(1+(Lh))^n \le e^{(Lhn)}$ and Equation (7) becomes

$$|E_n(x)| \le (e^{LT_f} - 1) \cdot \delta_x/(L \cdot h(p)). \tag{9}$$

In the rest of the paper, given an interval b, we use $l(b)$ and $r(b)$ to denote the lower

and the upper bounds of b, respectively. An interval b can be open on either or both sides; so, b can be $(l(b), r(b))$, $(l(b), r(b)]$, $[l(b), r(b))$, or $[l(b), r(b)]$. For arithmetic operations with $l(b)$ and $r(b)$, we assume that $\infty \pm x = \infty$ for any real x.

Definition 8. (IHA). *We assume that the HA is simulated for up to time T_f. So, the local truncation error can be accumulated only from time 0 to T_f. Given an HA A, an instrumented hybrid automaton of A, called IHA, is defined as a tuple $B = (A, N, h, T_f)$,* where

- $N : P_A \rightarrow P$ *ROG assigns to $p \in P_A$ a numerical method program with a stepsize $h(p)$,*
- $h : P_A \rightarrow \mathbb{R}^+$ *assigns to $p \in P_A$* a stepsize $h(p)$,
- *T_f is final time of the interval for which B runs,*
- *given A, N, h, and T_f, we compute β at the position p as follows:*

$$\forall v \in VC_A, \forall p \in P_A,$$
$$\beta_{p,x} = (e^{LT_f} - 1) \cdot \delta_x / (L \cdot h(p)),$$
$$\beta = max(\beta_{p,x}). \tag{10}$$

- *for each $p \in P_A$ and the invariant interval $I_{A,x}(p)$, $l(I_{B,x}(p)) = l(I_{A,x}(p)) + \beta$, $r(I_{B,x}(p)) = r(I_{A,x}(p)) - \beta$, for all $x \in VC_A$, and*
- *for each $e \in E_A$ and the guard interval $G_{A,x}(e)$, $l(G_{B,x}(e)) = l(G_{A,x}(e)) + \beta$, $r(G_{B,x}(e)) = r(G_{A,x}(e)) - \beta$, for all $x \in VC_A$.*

In general, checking numerically whether an invariant is violated or not can be reduced to a numerical event detection problem. That is, we can detect an invariant violation using various numerical event detection algorithms. However, if the dynamics of an HA changes rapidly during the smallest stepsize $h(p)$, the occurrence of such an event cannot be detected. This can be true if there are also multiple occurrences of the event during the stepsize. For the problem of an event detection, refer to the articles [Esposito et al. 2001; Park and Barton 1996]. To exclude such an intractable HA, we define the term *h-insensitive* hybrid automata to denote the class of HA without such behavior. Our main theorem provided later is on the soundness of IHA for *h-insensitive* HA.

Definition 9. (*h-insensitive* HA). *Given an HA A, let $B = (A, N, h, T_f)$ be an IHA, then the invariant $I_{A,x}(p)$ is called $h_B(p)$-insensitive if, for some t' such that $t < t' \leq t + h_B(p)$, the value of x at time t' is not in $I_{A,x}(p)$ then for any $t'' \in [t', t + h_B(p)]$ the value of x at t'' is not in $I_{A,x}(p)$, where $t = 0, h_B(p), 2 \cdot h_B(p), ..., \lfloor T_f / h_B(p) \rfloor \cdot h_B(p)$. When all invariants in HA A are h_B-insensitive, we call A h_B-insensitive HA.*

There are two major factors for simulating hybrid automata correctly: event detection and numerical errors. Since this paper focuses on the effect of numerical errors during simulation of hybrid automata, HA given in this paper is always assumed to be h_B-insensitive. By assuming h_B-insensitivity of hybrid automata, we can guarantee that there is no invariant violation during a time interval of which start and end points satisfy the invariant condition as stated below.

Lemma 1. *Given an HA A, let* $B = (A, N, h, T_f)$ *be an IHA, if A is* h_B*-insensitive and the both values of x at time t and* $t + h_B(p)$ *are in* $I_{A,x}(p)$*, then the value of x at* $t \in [t, t + h_B(p)]$ *is always in* $I_{A,x}(p)$.

Proof. Immediately followed by Definition 9.

After translating an HA A into its IHA B, the numbers of positions and transitions in an instrumented hybrid automaton can be different from those in a given hybrid automaton. That is, if we view the positions and transitions of HA A as a directed graph, IHA B is not necessarily isomorphic to a given HA A. If for some $x \in VC_A$, $l(I_{B,x}(p_i)) > r(I_{B,x}(p_i))$, then the invariant of p_i can never be satisfied, and thus, the position p_i is removed. Also, there are two cases, called *disabled* and *illegitimate*, in which a transition can never be taken. We say that the transition (p_i, p_j) is *disabled* if for some $x \in VC_A$, $l(G_{B,x}((p_i, p_j))) > r(I_{B,x}(p_i))$. We say that the transition (p_i, p_j) is *illegitimate* if for some $x \in VC_A$, $R_B((p_i, p_j), x) \notin I_{B,x}(p_j)$. We delete every transition that is *disabled* or *illegitimate*. Also, if all the outgoing edges of p_i disappear, we eliminate p_i in the IHA B.

Figure 3 shows an instrumented version of HA *Therm*. To instrument the HA *Therm*, we first compute, β_{p_0}, β_{p_1}, and β_{p_2}. If we use Euler's method for $t \in [0,3.0]$ with the integration stepsize 0.001, then $\beta_{p_0} = |2h^2(p_0) \times \lfloor T_f / h(p_0) \rfloor| = 0.006$. Likewise, we get $\beta_{p_1} = |(e^{3.0} - 1) \times e^{3.0} \times h\frac{h(P_2)}{2} = 0.192$ and $\beta_{p_2} = 0$. Therefore, β is $max(\beta_{p_0}, \beta_{p_1}, \beta_{p_2}) = 0.192$. In HA *Therm*, we have found that the guard of $G_{Therm}((p_0, p_2))$ becomes $x > 50$ by numerical error. Therefore, that faulty transition was caused. In IHA *Therm*, however, we can guarantee sound simulation because the guard of $G_{Therm}((p_0, p_2))$ becomes $x > 50 + \beta = 50.192$.

The definitions of *state* and *discrete transition step* for IHA are the same as the definitions in Section 2.

Definition 10. (Continuous transition step of an IHA). *Given an IHA B, a pair of states* (s_i, s_j) *is called a unit continuous transition step if the following conditions are satisfied:*

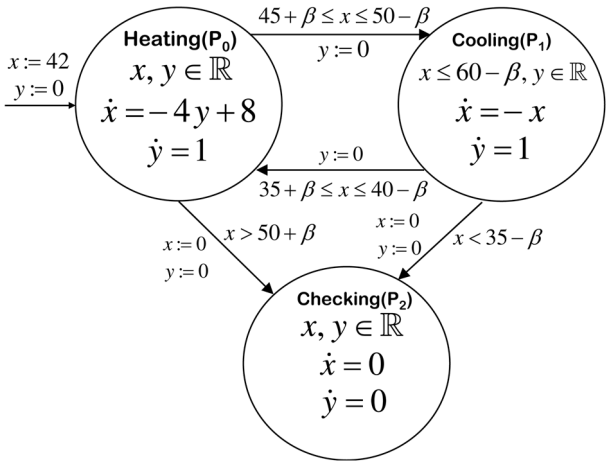


Figure 3. Instrumented Hybrid Automaton of Thermostat IHA_T.

– $s_j|t = s_i|_t + h(s_i|_p)$,
– $s_i|_p = s_j|_p$,
– *for all* $i, j, s_i|u, s_j|_u \in I_A(s_i|_p)$, *and*
– *for all* $x \in VC$, $s_j|_u(x)$ *is computed with* $N(s_j|_p)$, $h(s_j|p)$, and $s_i|_u(x)$.

When $\langle(s_i, s_{i+1}), (s_{i+1}, s_{i+2}),..., (s_{j-1}, s_j)\rangle$ *is a sequence of unit continuous transition steps, we say that* (s_i, s_j) *is a continuous transition step. When it is clear, a unit continuous transition step is also called a continuous transition step.*

Definition 11. (Run of an IHA). *Given an HA A, a run of the IHA B is a finite sequence of states,* $\langle s_0, s_1,..., s_i, s_{i+1},..., s_m\rangle$, *such that*

– $s_m|_t \le T_f$,
– $s_0 = (p_0, INIT_A, 0)$,
– *for all* $i \ge 0$, $s_{i+1}|_t$ *is either* $s_i|_t$ *or* $s_i|_t + h(s_i|_p)$,
– *if* $s_{i+1}|_t = s_i|_t$, (s_i, s_{i+1}) *is a discrete transition step at time* $s_i|_t$, *and*
– *if* $s_{i+1}|_t = s_i|_t + h(s_i|_p)$, (s_i, s_{i+1}) *is a continuous transition step at time* $s_i|_t$.

Similar to the alternating run of an HA defined in Definition 7, we merge multiple continuous transition steps into one continuous transition step, yielding the following definition of the alternating run of an IHA. The alternating run of an IHA is a finite sequence of states while that of an HA can be an infinite sequence.

Definition 12. (Alternating run of an IHA). *The alternating run of IHA B,* $\langle s_0,..., s_i, s_{i+1},..., s_m\rangle$, *is defined by a sequence of states which satisfies the following conditions:*

– *for* $i = 1...m-2$, *if* (s_i, s_{i+1}) *is a discrete transition step, then* (s_{i+1}, s_{i+2}) *is a continuous transition step and*
– *for* $i = 0...m-2$, *if* (s_i, s_{i+1}) *is a continuous transition step, then* (s_{i+1}, s_{i+2}) *is a discrete transition step.*

Lemma 2. *Given an HA A, let* $B = (A, N, h, T_f)$ *be an IHA and let* s^A *and* s^B *be the states of A and B respectively, satisfying the conditions:*

– $s^A|_p = s^B|p$,
– $s^A|_u(x) = s^B|_u(x)$ *for all* $x \in VC$, *and*
– $s^A|_t = s^B|_t$.

For some s'^B *of B such that* $s'^B|_t = s^B|_t + c \times h(s^B|p)$ *with some positive integer c if A is* h_B*-insensitive and* (s^B, s'^B) *is a continuous transition step, then there exists a continuous transition step* (s^A, s'^A) *for some state* s'^A *in A, which satisfies following conditions:*

(1) $s'^A|_u(x) \in [s'^B|_u(x) - \beta, s'^B|_u(x) + \beta]$ *for all* $x \in VC$,
(2) $s'^A|_u(x) \in I_x(s'^A|\mathrm{p})$ for all $x \in VC$,
(3) $s'^A|_p = s'^B|_p$, *and*
(4) $s'^A|_t = s'^B|_t$.

Proof. Let u be given by the solution of differential equations with the vector field $F(s^A|_p)$ at time $s^A|_t + c \times h(s^A|_p)$ and let the state $(s^A|_p, u, s^A|_t + c \times h(s^A|_p))$ be s'^A.

First, we show that the first condition is satisfied. Let x be a variable in VC. Since the value $s'^B|_u(x)$ is the result of numerical integration for c steps with the numerical method $N(s^B|_p)$, $s'^A|_u(x)$ is bounded in $s'^B|_u(x) \pm \beta'$, where β' is the global error from $s^B|_t$ to $s'^B|_t$. By Definition 8 we know that $\beta' \leq \beta$. Hence, we get $|s'^A|_u(x) - s'^B|_u(x)| \leq \beta$. Hence, $-\beta \leq s'^A|_u(x) - s'^B|_u(x) \leq \beta$, which yields that $s'^A|_u(x) \in [s'^B|_u(x) - \beta, s'^B|_u(x) + \beta]$. We conclude, therefore, that $s'^A|_u(x) \in [s'^B|u(x) - \beta, s'^B|_u(x) + \beta]$ for all $x \in VC$ ⋯ (1).

Second, we show that the invariants are not violated. Let x be a variable in VC. We know that $s'^B|_u(x) \in [l(I_x(s^A|_p)) + \beta, r(I_x(s^A|_p)) - \beta]$, since the invariants are not violated at time $s'^B|_t$. Using (1), we can get $s'^A|_u(x) \in [l(I_x(s^A|_p)), r(I_x(s^A|_p))]$, that is, $s'^A|_u(x) \in I_x(s^A|_p)$ ⋯ (2).

Third, we show that the positions are identical. Since $s'^B|_p = s^B|_p$ and $s'^A|_p = s^A|_p$, and by the given condition $s^B|_p = s^A|_p$, we have $s'^A|_p = s'^B|_p$ ⋯ (3).

Fourth, we show that the points of time are identical. Since $s'^B|_t = s^B|_t + c \times h(s^B|_p)$, and $s^A|_t = s^B|_t$, we have $s'^A|_t = c \times h(s^A|_p)$. Note that $s^A|_p = s^B|_p$.

Therefore, $s'^A|_t = s'^B|_t$ ⋯ (4).

Finally, since the HA A is h_B-insensitive, we know that the value of x is in $I_x(s^A|_p)$, for every $t' \in [s^A|_t, s'^A|_t]$, by the conditions $s^A|_u(x), s'^A|_u(x) \in I_x(s^A|_p)$, and Lemma 1 ⋯ (5).

From (1), (2), (3), (4), and (5), we get the transition step (s^A, s'^A) is a continuous transition step satisfying the conditions.

Lemma 3. *Given an HA A, let $B = (A, N, h, T_f)$ be an IHA and suppose there are states s^A and s^B that satisfy the following conditions:*

- $s^A|_p = s^B|_p$,
- $s^A|_u(x) \in [s^B|_u(x) - \beta, s^B|_u(x) + \beta]$ *for all* $x \in VC$,
- $s^A|_t = s^B|_t$, *and*
- $s^B|_u(x) \in G_x(s^B|_p)$.

Then, for some state s'^B of B, if (s^B, s'^B) is a discrete transition step then there exists a discrete transition step (s^A, s'^A) for some state s'^A in A, which satisfies the following conditions:

(1) $s'^A|_p = s'_B|_p$,
(2) $s'^A|_u(x) = s'^B|_u(x)$ *for all* $x \in VC$,
(3) $s'^A|_t = s'^B|_t$, *and*
(4) $s'^A|_u(x) \in I_x(s'^A|_p)$ *for all* $x \in VC$.

Proof. Let e^B be the edge such that $e^B = (p_0, p_1)$ in B, then since $E_B \subseteq E_A$, A has an edge $e^A = (p_0, p_1)$ such that $p_0 = s^B|_p$ and $p_1 = s'^B|_p$. It is easy to see that $p_0 = s^A|_p$, since $s^A|_p = s^B|_p$.

Let x be any variable such that $s^B|_u(x) \in G_x(e^B)$. Then, since $s^B|_u(x) \in G_x(e^B)$, we have $s^B|_u(x) \in [l(G_x(e^A)) + \beta, r(G_x(e^A)) - \beta]$. Also, since $s^A|_u(x) \in [s^B|_u(x) - \beta, s^B|_u(x) + \beta]$, we know that $s^A|_u(x) \in [l(G_x(e^A)) + \beta - \beta, r(G_x(e^A)) - \beta + \beta]$, which is $s^A|_u(x) \in$

$[l(G_x(e^A)), r(G_x(e^A))]$. Hence, $s^A|_u(x) \in G_x(e^A)$, meaning that the edge e^A is enabled in the state s^A.

Since the reset conditions in e^A are the same as those in e^B, we know that after the transition e^B is taken the value of every $x \in VC$ is identical to that of $s'^B|_u(x)$. Let the state s'^A in A be $(p_1, s'^A|_u, s^A|_t)$. Then we have that $s'^A|_p = s'^B|_p$, since $s'^A|_p = p_1 = s'^B|_p \cdots$ (1). Also, we know that for all $x \in VC$, $s'^A|_u(x) = s'^B|_u(x) \cdots$ (2). Furthermore, $s'^A|_t = s'^B|_t$ since $s'^A|_t = s^A|_t = s^B|_t = s'^B|_t \cdots$ (3).

Now we show that the invariant condition in $s'^A|_p$ is not violated. By the invariant condition in $s'^B|_p$, we know that $s'^B|_u(x) \in I_x(s'^B|_p)$ for all $x \in VC$, which is, by definition, $s'^B|_u(x) \in [l(I_x(s'^A|_p)) + \beta,\ r(I_x(s'^A|_p)) - \beta]$ for all $x \in VC$. Hence, having $s'^B|_u(x) = s'^A|_u(x)$ for all $x \in VC$, $s'^A|_u(x) \in [l(I_x(s'^A|_p)) + \beta,\ r(I_x(s'^A|_p)) - \beta]$ for all $x \in VC$. Therefore, $s'^A|_u(x) \in I_x(s'^A|_p)$ for all $x \in VC \cdots$ (4).

By (1), (2), (3), and (4), a discrete transition step (s^A, s'^A) exists and it satisfies the conditions.

Theorem 1. *Given an HA A, let* $B = (A, N, h, T_f)$ be an IHA, *such that A is* h_B*-insensitive. Then, for every alternating run* $\langle s_0^B, ..., s_i^B, ..., s_m^B \rangle$, *there exists an alternating run* $\langle s_0^A, ..., s_i^A, ..., s_m^A \rangle$ *of A such that for all* $0 \le i \le m$, *where* $s_m^B|_t \le T_f$:

- $s_i^A|_p = s_i^B|_p$,
- $s_i^A|_u(x) \in [s_i^B|_u(x) - \beta,\ s_i^B|_u(x) + \beta]$ *for all* $x \in VCA$, *and*
- $s_i^A|_t = s_i^B|_t$.

Proof. Immediately followed by Lemma 2 and Lemma 3.

4. CONCLUSION

This paper presented a method for taming the effect of numerical errors during simulation of hybrid automaton. As illustrated in the paper, numerical errors can result in allowing transitions that are not possible in the original model. Such anomalies have been observed in simulation-based tools including Simulink/Stateflow. The contribution of the paper is to identify the sources of numerical errors, to determine bounds on them, to use the bounds for instrumenting the guards and invariants of the original hybrid automaton so that impossible behavior cannot occur during simulation. More specifically, with the definitions of state, run, and alternating run of IHA, we show that the alternating run of IHA is always included in HA.

In this paper, we assume the error bounds are static and applied to the all states. This results in rather pessimistic IHA. So, we are currently extending the approach as follows. First, instead of using β, it should be possible to instrument guards and invariants based on position-specific β_p for each position p. Furthermore, it may be possible to get a tighter β_p for each position p if it determined during simulation. Second, if we are given a bound on the largest allowable local error, the bound can be used to determine the smallest k-th RK method that guarantees the bound. Third, Theorem 1 captures soundness of IHA with respect to HA, using the notions of soundness and completeness in logic [Gallier 1986]. It should be possible to achieve a (relative) completeness of IHA by examining how the stepsize in.uences the runs of

IHA, and also our approach can be extended to sound code generation framework of hybrid model, like [Hur et al. 2004].

ACKNOWLEDGMENT

This work was partially supported by the Engineering Research Center of Excellence Program of Korea Ministry of Education, Science and Technology (MEST)/Korea Science and Engineering Foundation (KOSEF), grant number R11-2008-007-03002-0.

REFERENCES

ABATE, A., A. AMES, AND S. SASTRY. 2006. Error bounds based stochastic approximations and simulations of hybrid dynamical systems. *American Control Conference, 2006*, 6.

ALUR, R., C. COURCOUBETIS, N. HALBWACHS, T. A. HENZINGER, P.-H. HO, X. NICOLLIN, A. OLIVERO, J. SIFAKIS, AND S. YOVINE. 1995. The algorithmic analysis of hybrid systems. *Theoretical Computer Science 138*, 1 (Feb.), 3–34.

ALUR, R. AND D. L. DILL. 1994. A theory of timed automata. *Theoretical Computer Science 126*, 2, 183–235.

CHUTINAN, A. 1999. Hybrid system verification using discrete model approximations. Ph.D. thesis, pages 11–13. Department of Electrical and Computer Engineering, Carnegie Mellon University.

CHUTINAN, A. AND B. KROGH. 2003. Computational techniques for hybrid system verification. *IEEE Transactions on Automatic Control 48*, 1 (JANUARY), 64–75.

DONZE, A. AND O. MALER. 2007. Systematic simulation using sensitivity analysis. In *HSCC*. 174–189.

ESPOSITO, J. M., V. KUMAR, AND G. J. PAPPAS. 2001. Accurate event detection for simulating hybrid systems. In *Hybrid Systems: Computation and Control*, M. D. D. Benedetto and A. L. Sangiovanni-Vincentelli, Eds. LNCS 2034. Springer, 204–217.

FIERRO, R. B., A. K. DAS, J. SPLETZER, Y. HUR, R. ALUR, J. M. ESPOSITO, G. Z. GRUDIC, V. KUMAR, I. LEE, J. P. OSTROWSKI, G. J. PAPPAS, J. SOUTHALL, AND C. J. TAYLOR. 2002. A framework and architecture for multirobot coordination. *International Journal of Robotics Research 10-11*, 977–995.

GALLIER, J. H. 1986. *Logic for computer science: Foundations of automatic theorem proving*. Harper & Row, New York, US.

GEAR, C. W. 1971. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentiec-Hall, Englewood Cli.s, US.

HENZINGER, T. 1996. The theory of hybrid automata. In *Proceedings of the 11th Annual Symposium on Logic in Computer Science*. IEEE Computer Society Press, 278–292.

HENZINGER, T. A., B. HOROWITZ, R. MAJUMDAR, AND H. WONG-TOI. 2000. Beyond HYTECH: Hybrid systems analysis using interval numerical methods. In *HSCC*. 130–144.

HICKEY, T. J. AND D. K. WITTENBERG. 2004. Rigorous modeling of hybrid systems using interval arithmetic constraints. In *Hybrid Systems:Computation and Control*. Springer Press, 402–416.

HUR, Y., J. KIM, I. LEE, AND J.-Y. CHOI. 2004. Sound code generation from communicating hybrid models. In *Hybrid Systems: Computation and Control*. Springer Press, 432–447.

JAIN, M. K. 1979. *Numerical Solution of Differential Equations*. John Wiley & Sons, New York, US.

LAFFERRIERE, G., G. J. PAPPAS, AND S. YOVINE. 1999. A new class of decidable hybrid systems. In *Hybrid Systems: Computation and Control*, F. W. Vaandrager and J. H. van Schuppen, Eds. LNCS 1569. Springer, 137–151.

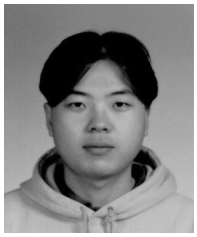
LYNCH, N. A., R. SEGALA, F. W. VAANDRAGER, AND H. B. WEINBERG. 1995. Hybrid I/O automata. In *Hybrid Systems III*, R. Alur, T. A. Henzinger, and E. D. Sontag, Eds. LNCS 1066. Springer, 496–510.

PARK, T. AND P. BARTON. 1996. State event location in differential-algebraic models. *ACM*

Transactions on Modeling and Computer Simulation 6, 2 (April), 137–165.

PRESS, W. H., S. A. TEUKOLSKY, W. T. VETTERLING, AND B. P. FLANNERY. 1999. *Numerical Recipes in C: the Art of Scientific Computing, 2nd edition*. Cambridge University Press, Cambridge, UK.

Yerang Hur received his BS and MS degrees in Computer Engineering 1994 and 1996, respectively from Seoul National University, Korea. His research interst includes embedded system design, modeling and analysis of hybrid systems, parallel and distributed simulation, real-time communication, and QoS support for broadband communication. He is a Ph.D. candidate of the Department of Computer and Information Science at the University of Pennsylvania and currently, he is employed by Posdata America R&D center.



Jae-Hwan Sim received the B.S degree from Yonsei University, Seoul, Korea, in 2002, the M.S. degree from Korea University, Seoul, Korea, in 2006. He is currently in Ph.D. course at Korea University. His research interests include real-time embedde system, formal methods, and control system.

Jesung Kim received the BS, MS, and PhD degrees in computer engineering from Seoul National University, Korea, in 1991, 1993, and 1998, respectively. He pursued postdoctoral research at the University of Pennsylvania and at Seoul National University. He was a research engineer at the Information & Telecommunications R&D Center of Hyundai Electronics, Korea, from 1998 to 2000. Currently, he is with The MathWorks, Inc., where he has been a Senior Software Developer since 2005. His research interests include model-based embedded systems design, hybrid systems, computer architecture, memory management, and Bluetooth-based personal area networking.

Jin-Young Choi received the B.S. degree from Seoul National University, Seoul, Korea, in 1982, the M.S. degree from Drexel University in 1986, and the Ph.D. degree from University of Pennsylvania, in 1993. He is currently a professor of Computer Science and Engineering Department, Korea University. His research interests are in real-time computing formal methods, security, software engineering, and protocol engineering.