Proceedings of the 36th
Conference on Decision & Control
San Diego, California USA • December 1997

Viability of Controllers for Hybrid Machines

Michael Heymann [†], *Feng Lin* [‡] and *George Meyer* [§]

Abstract

In this paper, we study the control of *Composite Hybrid Machines* (CHMs) subject to configuration-based safety specifications. CHMs are a class of hybrid systems modeled in modular fashion as the concurrent operation of *Elementary Hybrid Machines* (EHMs). We recall the algorithm introduced in [14], [15], [16] for synthesis of *minimally-interventive* controllers that guarantee constraint satisfaction. The paper focuses on essential questions associated with viability of a synthesized controller as related to the possibility of Zenoness of the controlled system. A hybrid system is Zeno if it can undergo an unbounded number of transitions in a bounded length of time.

1 Introduction

Various formalisms have been proposed in the literature to capture the basic structure of hybrid systems as dynamical systems in which discrete and continuous behaviors interact [2] [3] [8] [9] [17] [20]. In recent years interest in hybrid systems and their control has been rapidly growing (see e.g. [10], [6], [4], [19]).

Typical hybrid systems interact with their environment both by sharing signals (i.e., by transmission of input/output data), and by event synchronization (through which the system is reconfigured and its structure modified). Thus control of hybrid systems can be achieved by employing both interaction mechanisms simultaneously. Yet, while this flexibility adds significantly to the potential control capabilities, it clearly makes the problem of design very difficult.

A simplified interaction mechanism for control, is to allow the controller to interact with the system only discretely, that is, to permit the controller to trigger only discrete changes in the system. Synthesis of controllers of this type for restricted classes of hybrid systems recently received significant attention in the literature [7], [12], [14], [15], [16], [18], [21]. In [7] and [18] attention was focused on controller synthesis for timed automata [1], and emphasis was placed on the question of solvability. It was shown there that for timed automata the synthesis problem is solvable (that is, finite termination is assured). In [12] and [21] the decidability question was explored for *linear* and *rectangular* hybrid automata and it was shown that although the synthesis problem is generally undecidable (see e.g. [11]), under special conditions such as finite delay, rectangular automata are decidable and synthesis algorithms are guaranteed to terminate.

In [14], [15], [16], we examined the control problem for a class of hybrid systems that we called *rate-bounded* Composite Hybrid Machines (CHMs). We described a detailed algorithm for synthesis of a minimally interventive safety controller (that prevents the system from ever entering a specified set of illegal configurations). A (legal) controller is minimally interventive if, when composed to operate concurrently with any other legal controller, it will remain inactive except at the boundary of the legal region where controller inaction might lead to inevitable safety violation.

A major difficulty associated with hybrid systems is the Zenoness phenomenon. Intuitively, a system is Zeno if it can undergo an unbounded number of discrete changes (transitions) in a bounded length of time. When a controlled CHM is Zeno, it cannot be guaranteed to satisfy the safety specification indefinitely, and hence violates the basic *viability* requirement. If for a given CHM no legal controller exists such the controlled system is non-Zeno, we must conclude that the CHM is uncontrollable. The computational verification of non-Zenoness has been shown to be algorithmically a hard problem [5].

In the present paper we continue the investigation of [14], [15], [16] and focus our attention on the problem of Zenoness. In particular, we show that when our controller synthesis algorithm terminates, then the synthesized controller is legal and minimally interventive if the resultant closed loop system is non-Zeno. To examine the non-Zenoness, we introduce two concepts: Instantaneous Configuration Clusters (ICCs) and Hybrid Attractors. We then show that the system is non-Zeno if and only if it has no ICC that is a hybrid attractor.

2 Hybrid Machines

We first introduce a modeling formalism for a class of hybrid systems which we call *hybrid machines.* An elementary hybrid machine (EHM) is denoted by

$$EHM = (Q, \Sigma, D, I, E, (q_0, x_0)).$$

The elements of EHM are as follows.

Q is a finite set of vertices.

Σ is a finite set of event labels. An event is an input event, denoted by $\underline{\sigma}$ (underline), if it is received by the EHM from its environment; and an output event,

[†]Department of Computer Science, Technion, Israel Institute of Technology, Haifa 32000, Israel, e-mail: heymann@cs.technion.ac.il. The work by this author was completed while he was a Senior NRC Research Associate at NASA Ames Research Center, Moffett Field, CA 94035.

[‡]Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202, e-mail: flin@ece.eng.wayne.edu.

[§]NASA Ames Research Center, Moffett Field, CA 94035, e-mail: meyer@tarski.arc.nasa.gov.

denoted by $\overline{\sigma}$ (overline), if it is generated by the EHM and transmitted to the environment.

$D = \{d_q = (x_q, y_q, u_q, f_q, h_q) : q \in Q\}$ is the dynamics of the EHM, where d_q, the dynamics at the vertex q, is given by:

$$\dot{x}_q = f_q(x_q, u_q),$$
$$y_q = h_q(x_q, u_q),$$

with x_q, u_q, and y_q, respectively, the state, input, and output variables of appropriate dimensions. f_q is a Lipschitz continuous function and h_q a continuous function. (A vertex need not have dynamics associated with it, that is $d_q = \emptyset$, in which case we say that the vertex is *static.*)

$I = \{I_q : q \in Q\}$ is a set of invariants. I_q represents conditions under which the EHM is permitted to reside at q. A formal definition of I_q will be given below.

$E = \{(q, G \wedge \underline{\sigma} \rightarrow \overline{\sigma'}, q', x^0_{q'}) : q, q' \in Q\}$ is a set of edges (transition-paths), where q is the exiting vertex, q' the entering vertex, $\underline{\sigma}$ the input-event, $\overline{\sigma'}$ the output-event, G the guard to be formally defined below, and $x^0_{q'}$ the initialization value for $x_{q'}$ upon entry to q'.

$(q, G \wedge \underline{\sigma} \rightarrow \overline{\sigma'}, q', x^0_{q'})$ is interpreted as follows: If G is true and the event $\underline{\sigma}$ is received as an input, then the transition to q' takes place with the assignment of the initial condition $x_{q'}(t_0) = x^0_{q'}$ (here t_0 denotes the time at which the vertex q' is entered and $x^0_{q'}$ is a vector of constants). The output-event $\overline{\sigma'}$ is transmitted at the same time.

If $\overline{\sigma'}$ is absent, then no output-event is transmitted. If $x^0_{q'}$ is absent, then the initial condition is inherited from x_q (assuming x_q and $x_{q'}$ represent the same physical object and hence are of the same dimension). If $\underline{\sigma}$ is absent, then the transition takes place immediately upon G becoming true. If G is absent, the guard is always true and the transition will be triggered by the input-event $\underline{\sigma}$.

(q_0, x_0) denote the initialization condition: q_0 is the initial vertex and $x_{q_0}(t_0) = x_0$.

For the EHM to be well-defined, we require that all vertices be *completely guarded.* That is, every invariant violation implies that some guard associated with a dynamic transition becomes true*. (It is, in principle, permitted that more than one guard become true at the same instant. In such a case the transition that will actually take place is resolved nondeterministically.) We do not require the converse to be true. That is, a transition can be triggered even if the invariant is not violated. We further require that, upon entry to a vertex q', the invariant $I_{q'}$ be true. It is however possible that, upon entry to q', one of the guards at q' is already true. In such a case, the EHM will immediately exit q' and enter a vertex specified by (one of) the true guards. Such a transition is considered instantaneous.

*Complete guardedness prevents the possibility that an invariant becomes false while no transition out of the current vertex is dynamically triggered.

In this paper we shall study a restricted class of hybrid machines called *bounded-rate* hybrid machines†, characterized by the following assumption.

Assumption 1 The dynamics described by f_q and h_q has the following properties: (1) $h_q(x_q, u_q)$ is a linear function; and (2) $f_q(x_q, u_q)$ is bounded by a lower limit k^L_q and an upper limit k^U_q; that is, the only information given about $f_q(x_q, u_q)$ is that $f_q(x_q, u_q) \in [k^L_q, k^U_q]$.

A composite hybrid machine (CHM) consists of several elementary hybrid machines running in parallel:

$$CHM = EHM^1 || EHM^2 || ... || EHM^n.$$

Interaction between EHMs is achieved by means of signal transmission (shared variables) and input/output-event synchronization (message passing) as described below.

Shared variables consist of output signals from all EHMs as well as signals received from the environment. They are shared by all EHMs in the sense that they are accessible to all EHMs. A shared variable s_i can be the output of at most one EHM. The set of shared variables defines a signal space $S = [s_1, s_2, ..., s_m] \in \mathcal{R}^m$.

Transitions are synchronized by an input/output synchronization formalism. That is, if an output-event $\overline{\sigma}$ is either generated by one of the EHMs or received from the environment, then all EHMs for which $\underline{\sigma}$ is an active transition label (i.e., $\underline{\sigma}$ is defined at the current vertex with a true guard) will execute $\underline{\sigma}$ (and its associated transition) concurrently with the occurrence of $\overline{\sigma}$. A specific output-event can be generated by at most one EHM. Clearly, input-events do not synchronize among themselves‡.

By introducing the shared variables S, we can now define invariants and guards formally as boolean combinations of inequalities of the form (called *atomic formulas*)

$$s_i \geq C_i \quad \text{or} \quad s_i \leq C_i,$$

where s_i is a shared variable and C_i is a real constant. We shall assume that all invariants are closed (that is, the sets in which the invariants are true are closed).

To describe the behavior of

$$CHM = EHM^1 || EHM^2 || ... || EHM^n,$$

we define a *configuration* of the CHM to be

$$q = < q^1_{i_1}, q^2_{i_2}, ..., q^n_{i_n} > \in Q^1 \times Q^2 \times ... \times Q^n,$$

where Q^j is the set of vertices of EHM^j (components of the EHMs are superscripted).

†Similar classes of hybrid automata [2] have been termed "rectangular".

‡Notice that this formalism is a special case of the prioritized synchronous composition formalism [13], where each event is in the priority set of at most one parallel component.

A transition

$$< q^1_{i_1}, q^2_{i_2}, ..., q^n_{i_n} > \xrightarrow{l} < q^1_{i'_1}, q^2_{i'_2}, ..., q^n_{i'_n} >$$

of a CHM is a triple, where $q =< q^1_{i_1}, q^2_{i_2}, ..., q^n_{i_n} >$ is the source configuration, $q' =< q^1_{i'_1}, q^2_{i'_2}, ..., q^n_{i'_n} >$ the target configuration, and l the label that triggers the transition. l can be either an event, or a guard becoming true. Thus, if $l = \underline{\sigma}$ is an event (generated by the environment), then either $q^j_{i'_j} = q^j_{i_j}$ if $\underline{\sigma}$ is not active at $q^j_{i_j}$, or $q^j_{i'_j}$ is such that $(q^j_{i_j}, \underline{\sigma} \rightarrow \overline{\sigma'}, q^j_{i'_j}, x^0_{q^j_{i'_j}})$ is a transition (edge) in E^j. On the other hand, if $l = G$ is a guard, then there must exist a transition $(q^m_{i_m}, G \rightarrow \overline{\sigma'}, q^m_{i'_m}, x^0_{q^m_{i'_m}})$ in some EHM^m, and for $j \neq m$, either $q^j_{i'_j} = q^j_{i_j}$ if $\underline{\sigma'}$ is not defined at $q^j_{i_j}$, or $q^j_{i'_j}$ is such that $(q^j_{i_j}, \underline{\sigma'} \rightarrow \overline{\sigma''}, q^j_{i'_j}, x^0_{q^j_{i'_j}})$ is a transition in E^j.

For brevity we shall sometimes denote the transition simply by (q, l, q'). Note that for simplicity, we do not specify the output events and initial conditions, since they are defined in the EHMs.

The transitions are assumed to occur instantaneously, and concurrent vertex changes in parallel components are assumed to occur exactly at the same instant (even when constituting a logically triggered finite chain of transitions).

A *run* of the EHM is a sequence

$$q_0 \xrightarrow{e_1,t_1} q_1 \xrightarrow{e_2,t_2} q_2 \xrightarrow{e_3,t_3} \ldots$$

where e_i is the ith transition and t_i is the time when the ith transition takes place. The trajectory of the run is the sequence of the vector time functions of the state variables:

$$x_{q_0}, x_{q_1}, x_{q_2}, \ldots$$

where $x_{q_i} = \{x_{q_i}(t) : t \in [t_i, t_{i+1})\}$.

Recall that our model also allows guarded event transitions of the form

$$q \xrightarrow{G \wedge \underline{\sigma}} q'.$$

However, since for the transition to take place the guard must be true when the event is triggered, a guarded event transition can be decomposed into

$$\begin{tikzcd} q_1 \arrow[r, shift left, "G"] & q_2 \arrow[l, shift left, "\neg G"] \arrow[r, "\underline{\sigma}"] & q', \end{tikzcd}$$

where q has been partitioned into q_1 and q_2, with $I_{q_1} = I_q \wedge \neg G$ and $I_{q_2} = I_q \wedge G$. Thus, transitions in CHMs can be classified into two types: (1) dynamic transitions, that are labeled by guards only, and (2) event transitions, that are labeled by events.

3 Control

Our goal is to design a controller that will ensure that the controlled (or closed-loop) system satisfies a prescribed set of safety specifications. More precisely, it will be required that the controlled system never enter a set of illegal (or bad) configurations denoted by Q_b. We call such a controller a *legal* controller.

Formally, a (legal) controller of a CHM is an EHM C running in parallel with the CHM. In this paper, we assume that the interaction between C and CHM is by means of input/output event synchronization only.

A legal controller C is said to be *less interventive* (or less restrictive) than another legal controller C' if every run permitted by C' is also permitted by C (a formal definition will be given below). A legal controller is said to be *minimally interventive* if it is less interventive than any legal controller.

To synthesize the minimally interventive legal controller for the CHM

$$CHM = (Q, \Sigma, D, I, E, (q_0, x_0)),$$

we employ an invariance algorithm [16] whose brief outline is as follows.

Initially, we let the "bad" configurations consist of the illegal ones:

$$BC := Q_b.$$

The remaining configurations may or may not be bad, so we classify them as *pending*:

$$PC := Q - Q_b.$$

Our synthesis algorithm examines each configuration in PC iteratively as described below. The configurations (or some "parts" of them) that pass the examination are then placed in the set of *new pending configurations*, which is initially empty:

$$NPC := \emptyset.$$

During the synthesis, each pending configuration is marked by its configuration-origin, which is initially set as

$$CO(q) = q.$$

The algorithm is iterative. During each iteration we partition, the invariant of every configuration $q \in PC$ into two subsets:

$$\begin{aligned} I_{q_1} := \quad & I_q \wedge ((\wedge_{(q,G,q') \in DT_b(q,BC)} pc(q, G, q')) \\ & \vee(\vee_{(q,\underline{\sigma},q') \in ET_g(q,BC)} sc(q, \underline{\sigma}, q'))), \\ I_{q_2} := \quad & I_q \wedge (\neg(\wedge_{(q,G,q') \in DT_b(q,BC)} pc(q, G, q')) \\ & \wedge\neg(\vee_{(q,\underline{\sigma},q') \in ET_g(q,BC)} sc(q, \underline{\sigma}, q'))). \end{aligned}$$

I_{q_1} represents the "pending part" of the configuration q, because when I_{q_1} is true, either a bad dynamic transition $(q, G, q') \in DT_b(q, BC)$ to a bad configuration will be preempted (since the preemptive condition $pc(q, G, q')$ is satisfied), or a good event transition $(q, \underline{\sigma}, q') \in ET_g(q, BC)$ to a pending configuration can be triggered (since the safe-exit condition $sc(q, \underline{\sigma}, q')$ is satisfied) [16].

I_{q_2}, the complement of I_{q_1}, represents the "bad part" of the configuration q.

If $I_{q_1} \neq false$, we set

$$NPC := NPC \cup \{q_1\},$$
$$CO(q_1) := CO(q).$$

If $I_{q_2} \neq false$, we set

$$BC := BC \cup \{q_2\}.$$

After examining all configurations in PC, we check whether the *stop-condition* PC=NPC is true. If not, we set

$$PC := NPC,$$
$$NPC := \emptyset.$$

and go to the next iteration.

When the stop-condition PC=NPC is satisfied, all configurations in PC are "good". Therefore, we can proceed to construct the minimally interventive legal controller

$$C = (Q^c, \Sigma^c, D^c, I^c, E^c, (q_0^c, x_0^c))$$

as follows.

$$Q^c := PC,$$
$$D^c := \emptyset,$$
$$I^c := I|_{Q^c}.$$

In other words, the configurations of C consist of the set of all "good" configurations with their invariants as calculated during the iteration. C has no continuous dynamics so it is "driven" by the CHM. The transitions of C are then triggered when the boundary of the invariant of the current configuration, as described by the critical condition *critical*(.) [16], is reached. Specifically,

$$\begin{aligned} E^c := \; & \{(q, critical(I_q) \wedge wp(q, \underline{\sigma}, q') \\ & \wedge(\neg(\wedge_{(q,G,q'') \in DT_b(q,BC)} pc(q, G, q''))) \to \overline{\sigma}, q') : \\ & q, q' \in Q^c \wedge (CO(q), \underline{\sigma}, CO(q')) \in E\}. \end{aligned}$$

The controller as just synthesized is minimally interventive. Its interaction with the system is restricted to the exclusive objective of preventing safety violation. Since other control objective are possible, we augment our controller by "environment-triggered" transitions given by

$$\begin{aligned} E^c := \; & E^c \cup \{(q, wp(q, \underline{\sigma}, q') \wedge \underline{\tilde{\sigma}} \to \overline{\sigma}, q') : \\ & q, q' \in Q^c \wedge (CO(q), \underline{\sigma}, CO(q')) \in E\} \end{aligned}$$

with the event set Σ^c being defined as

$$\Sigma^c := \; \Sigma \cup \{\underline{\tilde{\sigma}} : \sigma \in \Sigma\}.$$

Thus, the events $\tilde{\sigma}$ are allowed to be generated by the environment (possibly by an additional controller) and trigger transitions in C and hence in CHM whenever such transitions are not preempted by C. A preemption will occur only if otherwise a safety constraint would be violated.

4 Zenoness

We shall call a run of a CHM *dynamic* if all its transitions are dynamic transitions. An unbounded dynamic run

$$q_0 \xrightarrow{e_1, t_1} q_1 \xrightarrow{e_2, t_2} q_2 \xrightarrow{e_3, t_3} \ldots$$

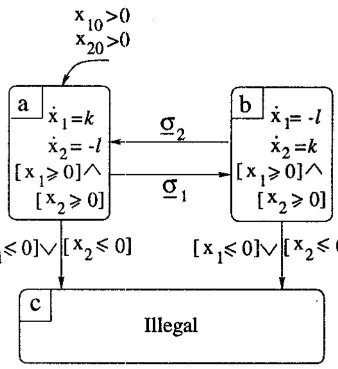
is called a *Zeno* run if

$$lim_{i \to \infty} t_i = T < \infty$$

A CHM is called *Zeno* if it possesses Zeno runs. Otherwise it is called *viable* or *non-Zeno*.

Clearly Zeno CHMs are ill defined, in that they may uncontrollably execute an unbounded number of transitions in a finite (and bounded) time interval and thus describe systems whose lifetime is limited, contrary to our intention of modeling ongoing behaviors (that never terminate). To illustrate the Zenoness phenomenon, let us examine the following example.

Example 1 Consider the hybrid system modeled by the CHM shown in Figure 1, where *k* and *l* are constant real numbers.



$x_{10}>0$
$x_{20}>0$
a
$\dot{x}_1=k$
$\dot{x}_2=-l$
$[x_1 \geqslant 0] \wedge [x_2 \geqslant 0]$
b
$\dot{x}_1=-l$
$\dot{x}_2=k$
$[x_1 \geqslant 0] \wedge [x_2 \geqslant 0]$
$\underline{\sigma}_2$
$\underline{\sigma}_1$
$[x_1 \leqslant 0] \vee [x_2 \leqslant 0]$
$[x_1 \leqslant 0] \vee [x_2 \leqslant 0]$
c
Illegal

Figure 1: CHM of Example 1

It models a two-tank water system, where both tanks are leaking with rate *l*. A pump fills one of the tanks at rate *k+l*, and it can be switched between the two tanks (events $\underline{\sigma_1}$ and $\underline{\sigma_2}$). The system starts with both tanks non-empty ($x_1(0) = x_{10} > 0, x_2(0) = x_{20} > 0$). The system becomes illegal if and when the level in one of the tanks gets to be below zero[§]. This is represented by a transition to the illegal configuration *c* that has no dynamics and invariant "true". It is readily seen that the rate of change of the total volume of water in the system is independent of the pump-switching policy and is given by *k-l*. Thus, if *k-l*< 0, then at time no later that $T = (x_{10} + x_{20})/$*(k-l)* the total volume of water will become zero, so that the system must have become illegal no later than T. On the other hand, if *k-l*$\geq$ 0, the volume of water does not diminish and, as easily seen, a viable controller actually exists.

The controller synthesis algorithm can be carried out for arbitrary *k* and *l*, and it terminates after just one step.

[§]For technical reasons we denote the guards by weak inequalities rather than strict ones.

The (augmented) controller C generated by the algorithm is shown in Figure 2. The guard G_i triggering the event transition $\overline{\sigma_i}$ is calculated as follows.

$$G_1 = [x_1 \geq 0] \wedge [x_2 = 0], G_2 = [x_1 = 0] \wedge [x_2 \geq 0].$$

In other words, the controller switches the pump to a tank whose level reaches 0. If $k - l < 0$, then the switching becomes faster and faster, with infinite switching rate occurring after finite time. Thus, the closed-loop system $CHM||C$ is Zeno.

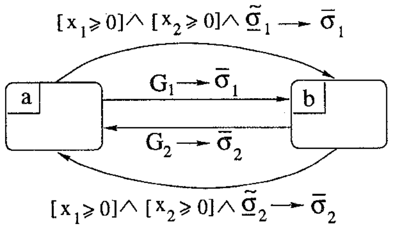

$[x_1 \geqslant 0] \wedge [x_2 \geqslant 0] \wedge \tilde{\sigma}_1 \rightarrow \bar{\sigma}_1$
a
$G_1 \rightarrow \bar{\sigma}_1$
b
$G_2 \rightarrow \bar{\sigma}_2$
$[x_1 \geqslant 0] \wedge [x_2 \geqslant 0] \wedge \tilde{\sigma}_2 \rightarrow \bar{\sigma}_2$

Figure 2: Controller of Example 1

To examine the Zenoness phenomenon and its relation to control synthesis, we begin by introducing the concept of an *instantaneous configuration cluster* (ICC). Let $v = [s_1, ..., s_m] \in S$ be a valuation of the signal vector and let q be a configuration. Suppose that q is entered by a guarded transition G whose value is true at v. Assume further that q has an outgoing guarded transition G' which becomes (or is) true at the entry value of the signal vector q. (This value will be v if the signal vector is not re-initialized, or it will be $v' \in S$ if it is re-initialized to this value upon entry to q.) Since G' follows G instantaneously, we say that the transition associated with G' is triggered by that associated with G. We say that a sequence of transitions $G_1, G_2, G_3, \ldots$ is triggered by v if G_1 is true at v and G_{i+1} is triggered by G_i for all $i \geq 1$. For a signal value v, consider all transition sequences in the CHM triggered by v. Let $CHM(v)$ denote the CHM obtained by deleting all transitions that are not elements of transition sequences triggered by v. A strongly connected component (SCC)[¶] that consists of two or more configurations is called an ICC. The triggering value v of the signal vector will be called a *Zeno point* of the CHM. We emphasize that a ICC consists of the set of configurations, the transitions and the associated triggering value v of the signal vector. In the two leaking tank example, $v = [0, 0]$ is a Zeno point associated with an ICC which includes the configurations $(a, a), (b, b)$ of the controlled system $CHM||C$.

The following theorem gives a necessary condition for Zenoness.

Theorem 1 If a CHM is Zeno, then there exists an instantaneous configuration-cluster.

The existence of an instantaneous configuration-cluster does not in itself imply Zenoness. In the two leaking tanks of Example 1, for instance, if $k - l > 0$ the closed-loop system is non-Zeno, although $v = [0, 0]$ is a Zeno point associated with an ICC.

[¶]A SCC is a set of configurations for which there is a directed path from any configuration to any other.

To obtain a necessary and sufficient condition for Zenoness, we shall now refine our examination of ICCs. Clearly, once at an ICC, the behavior of the CHM is necessarily Zeno. Thus, the question must be examined, whether if initialized outside (or away from) an ICC, a possible run will enter the ICC after a bounded length of time. We shall say that an ICC is a *hybrid attractor* whenever there exist initializations of the CHM outside the ICC such that for some run, the ICC will be reached in bounded time.

Theorem 2 A CHM is non-Zeno if and only if its initialization set does not intersect an ICC and its has no hybrid attractor.

Thus, the problem of checking Zenoness of a CHM (or, in particular, a closed-loop system) consists of identifying its ICCs, if any, and checking whether they include hybrid attractors. The detailed investigation of these issues will be presented elsewhere.

5 Correctness of the Algorithm

It is clear from Algorithm 1 that whenever a controlled systems $CHM||C$ undergoes a transition, it moves from a legal configuration to another. Thus, there remain two questions with respect to the correctness of the algorithm. The first is whether the controlled system is safe and viable. Since the safety is clear, this boils down to the question of non-Zenoness, which was discussed in the previous section. The second question is whether the synthesized controller is minimally interventive. To this end, we define the conjunction of C with another controller D as follows. First, all the output-events $\overline{\sigma}$ in D are replaced by $\overline{\overline{\sigma}}$ to obtain $\tilde{D}$. Then the composite controlled system is given by

$$CHM||C||\tilde{D}.$$

Theorem 3 If Algorithm 1 terminates in a finite number of steps and the closed-loop system $CHM||C$ is non-Zeno, then the controller synthesized is a minimally interventive legal controller in the following sense.

1. For any controller D (legal or not), a run of $CHM||C||\tilde{D}$ never visits illegal configurations in Q_b.
2. For any legal controller D, every run of $CHM||D$ has a corresponding run in $CHM||C||\tilde{D}$.

Proof

Since Algorithm 1 terminates in a finite number of steps and the closed-loop system $CHM||C$ is non-Zeno, it is safe and viable.

Notice that it is possible that the closed-loop system $CHM||C||\tilde{D}$ may generate a Zeno run due to $\tilde{D}$. Although such an ill-defined controller D should be avoided in practice, the correctness of C will not be affected.

To prove part 1, it is sufficient to show that a run in $CHM||C||\tilde{D}$ will only visit configurations in

$$Q^c \subseteq Q - Q_b.$$

If this is not the case, then there exists a run

$$q_0 \xrightarrow{e_1,t_1} q_1 \longrightarrow \ldots \longrightarrow q_{n-1} \xrightarrow{e_n,t_n} q_n$$

such that $q_0, q_1, \ldots, q_{n-1} \in Q^c$ but $q_n \notin Q^c$.

Let us consider the transition from q_{n-1} to q_n. It cannot be an event transition because such illegal event transitions are not permitted by C. If it is a dynamic transition, then since it is not preempted at q_{n-1}, it implies that $q_{n-1} \notin Q^c$, a contradiction.

To prove part 2, observe first that in view of the fact that Algorithm 1 progressively removes illegal behaviors, a controller will be legal only if it does not exceed the configurations and invariants of C. Assume that

$$q_0 \xrightarrow{e_1,t_1} q_1 \longrightarrow \ldots \longrightarrow q_{n-1} \xrightarrow{e_n,t_n} q_n$$

is a possible run of $CHM||D$ and the first $n-1$ transitions are also possible in $CHM||C||\tilde{D}$ but the last transition from q_{n-1} to q_n is impossible in $CHM||C||\tilde{D}$, that is, it is preempted by C. Since C only takes action at its invariant violation boundary as specified by the critical condition, the inaction of D at that point implies that for some trajectory associated with this run, the invariant of C will be violated, contradicting the hypothesis that D is legal.

References

[1] R. Alur and D. Dill, 1990. Automata for modeling real-time systems. *Proc. of the 17th International Colloquium on Automata, Languages and Programming*, pp. 322-336.

[2] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho, 1993. Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. *Hybrid Systems, Lecture Notes in Computer Science, 736*, Springer-Verlag, pp. 209-229.

[3] R. Alur, 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*, pp. 3-34.

[4] R. Alur, T. A. Henzinger, E. Sontag, (Eds.) 1996. *Hybrid Systems III, Verification and Control, Lecture Notes in Computer Science, 1066*, Springer Verlag.

[5] R. Alur and T.A. Henzinger, 1997. Modularity of timed and hybrid systems. *Preprint.*

[6] P. Antsaklis, W. Kohn, A. Nerode, S. Sastry, (Eds.) 1995. *Hybrid Systems II, Lecture Notes in Computer Science, 999*, Springer Verlag.

[7] E. Azarin, O. Maler, and A. Pnueli, 1995. Symbolic controller synthesis for discrete and timed systems, *Hybrid Systems II, Lecture Notes in Computer Science, 999*, Springer Verlag, pp. 1-20.

[8] M. S. Branicky, 1995. Universal computation and other capabilities of hybrid and continuous dynamical systems. *Theoretical Computer Science, 138*, pp. 67-100.

[9] R. W. Brockett, 1993. Hybrid models for motion control systems. *Essays in Control: Perspectives in the theory and its applications*, pp. 29-53, Birkhauser, Boston.

[10] R. L. Grossman, A. Nerode, Rischel, Raven, (Eds.) 1993. *Hybrid Systems, Lecture Notes in Computer Science, 736*, Springer Verlag.

[11] T. Henzinger, P. Kopke, A. Puri and P. Varaiya, 1995. What's decidable about hybrid automata. *Proc. of the 27th Annual ACM Symposium on the Theory of Computing.*

[12] T. A. Henzinger and P. W. Kopke, 1997. Discrete time control for rectangular hybrid automata. *Proceedings , 24th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science*, Springer Verlag.

[13] M. Heymann 1990. Concurrency and discrete event control. *IEEE Control Systems Magazine, Vol. 10, No. 4,* pp 103-112.

[14] M. Heymann, F. Lin and G. Meyer, 1997. Control synthesis for a class of hybrid systems subject to configuration based safety constraints. *Proceedings of HART97, Lecture Notes in Computer Science 1201* pp. 376-390, Springer Verlag.

[15] M. Heymann, F. Lin and G. Meyer, 1997. Synthesis of minimally restrictive controllers for a class of hybrid systems, to appear in *Hybrid Systems IV, Lecture Notes in Computer Science*, Springer Verlag.

[16] M. Heymann, F. Lin and G. Meyer, 1997. Control synthesis for a class of hybrid systems subject to configuration based safety constraints. *NASA Technical Memorandum 112196.*

[17] O. Maler, Z. Manna and A. Pnueli, 1991. From timed to hybrid systems. In *Real Time: Theory in Practice, Lecture Notes in Computer Science 600,* pp. 447-484. Springer-Verlag.

[18] O. Maler, A. Pnueli and J. Sifakis, 1995. On the synthesis of discrete controllers for timed systems. *Lecture Notes in Computer Science 900,* pp. 229-242. Springer-Verlag.

[19] O. Maler (Ed.) 1997. *Hybrid and Real-Time Systems, HART'97, Lecture Notes in Computer Science, 1201,* Springer Verlag.

[20] A. Nerode and W. Kohn, 1993. Models for hybrid systems: automata, topologies, controllability, observability. *Hybrid Systems, Lecture Notes in Computer Science, 736*, Springer-Verlag, pp. 317-356.

[21] H. Wong-Toi, 1997. Synthesis of controllers for linear hybrid automata, *Preprint.*