THE ANNALS OF "DUNAREA DE JOS" UNIVERSITY OF GALATI
FASCICLE III, 2000 ISSN 1221-454X
ELECTROTECHNICS, ELECTRONICS, AUTOMATIC CONTROL, INFORMATICS

COMMENTS ON CLOCK MODELS IN HYBRID AUTOMATA AND HYBRID CONTROL SYSTEMS

Virginia Ecaterina OLTEAN, Dorin CARSTOIU

"Politehnica" University of Bucharest, Faculty of Control and Computers
Spl.Independentei 313, sector 6, 77206 Bucharest, Romania
e-mail:{oltean,cid}@aii.pub.ro

Abstract: Hybrid systems have received a lot of attention in the past decade and a number of different models have been proposed in order to establish mathematical framework that is able to handle both continuous and discrete aspects. This contribution is focused on two models: hybrid automata and hybrid control systems with continuous-discrete interface and the importance of clock models is emphasized. Simple and relevant examples, some taken from the literature, accompany the presentation.

Keywords: hybrid systems, hybrid automaton, discrete event system (DES)

1. INTRODUCTION

Hybrid systems are systems that involve both continuous and discrete variables. Their evolution is given by state equations that generally depend on all the variables. The continuous dynamics are generally given by differential/difference equations. The discrete variables dynamics of hybrid systems are generally governed by an automaton or an input-output transition system with a finite number of states (Alur et. al, 1995), (Alam and Alla, 1998), (Favela, 1999). Hybrid systems have received a lot of attention in the past decade and a number of different models have been proposed in order to establish mathematical framework that is able to handle both continuous and discrete aspects. This contribution is focused on the evaluation of clock models in hybrid automata and in hybrid control systems with continuous –discrete interface.

The first class of models is dedicated to performance evaluation of hybrid systems. A hybrid automaton is basically a variable structure system, which consists of a number of continuous evolution laws - associated to the locations of the automaton - and a switching policy between these locations. Clocks are present in the model as continuous first order systems with constant speed, and they are used to govern the temporal conditions of the switching policy.
The second class of hybrid systems consists of a continuous plant that is governed, through an interface, by a DES controller. The plant is modeled by a set of differential equations and it receives a piecewise constant input signal from the controller. The continuous state space is partitioned, by smooth hypersurfaces, into open cells, and the continuous trajectory evolves from cell to cell, generating plant-events for the DES controller, as it crosses the hypersurfaces. The plant and the interface are first abstracted to a pure logic automaton, called the DES-plant. Then the controller is built as a Moore machine, by adapting the techniques from the Ramadge-Wonham DES theory (Ramadge and Wonham, 1989). The problem of clock models is related, in this second approach, to the implementation of the evolution equations of the DES controller.

The paper is organized as follows. In Section 2 a brief review of the hybrid automata models is presented and, starting from two relevant examples from the literature, the specific role of clocks is emphasized. Section 3 is dedicated to a brief overview of the models of hybrid control systems with interface, accompanied by the discussion of two

This paper was recomended for publication by S. Bumbaru

problems. Starting from the problem of a suitable rule-based implementation of the DES controller, by HORN clauses, a local discrete clock of the DES controller is introduced, and its relation to the global clock variable is emphasized. The second problem, discussed through a simple example, treats the introduction of a continuous time model of a clock, specific to the controller, when passing from a pure logic control task to a logic and temporal control task. Some preliminary conclusions are finally presented.

2. CLOCK MODELS IN HYBRID AUTOMATA

2.1 Review on hybrid automata and clock models

The framework used here is based on the hybrid automata first introduced in (Alur et. al, 1995). A hybrid system is modeled as a finite automaton that is equipped with a set of variables. In each location of the automaton, the values of the variables change continuously with time, according to a specific evolution law. Each transition of the automaton is guarded by an event and its execution modifies the values of the variables, according to the associated assignment. Each location is also labeled with an invariant condition that must hold as long as the system resides at a location.

Definition 1. A hybrid automaton H is defined as a seven-tuple $H = (Var, Loc, Act, Inv, A, Ev, Ass)$ where:

- *Var* is the finite number of real-valued variables x^1, x^2, …, x^n. The continuous state of H is $x = [x^1 \quad x^2 \quad \dots \quad x^n]^T \in \mathbf{R}^n$ and it is also the variable state of *H*..
- *Loc* is a finite set of vertices called *locations*;
- *Act* is a function that assigns to each location $l \in Loc$ a function act_l describing the evolution of variables according with time. In general, act_l can be defined as an evolution law:
$$act_l : \dot{x} = f(x, u)$$
- *A* is a finite set of arcs called *transitions*. Each transition $a = (l, l')$ identifies a source location $l \in Loc$ and a target location $l' \in Loc$.
- *Ev* is a function that assigns to each transition $a = (l, l')$ a predicate Ev_a, called event. The execution of the transition $a = (l, l')$ is conditioned by the occurrence of the event Ev_a.
- *Ass* is a function that assigns to each transition $a = (l, l')$ a relation Ass_a called *assignment*. It is used for updating the variable state and generally defined as:
$$Ass_a: x := g(x)$$

□

At any instant, the *state* of a hybrid system is given by a location *l* and a continuous (or variable) state *x*, so the system state is characterized by the pair (*l*,*x*). It can change in two ways:

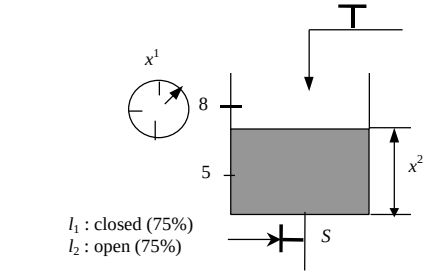
- by a *time delay* measured by a *clock*, that changes only the variable state according to the evolution law associated with current location; such a delay can take place as long as the variable state satisfies the location invariant;
- by the *instantaneous execution of a transition* that changes both the location and current variable state according to the transition assignment. The execution of the transition is conditioned by the occurrence of the associated event.

If in a location l $act_l : \dot{x} = 1$ and for all transitions having the source in l, $a = (l, l')$ with $l' \in Loc$, the assignment is defined by: Ass_a: $x := g(x) \in \{0, x\}$, then x is a *clock*. Thus: (1) the value of the clock increases uniformly with time, and (2) a discrete transition either resets a clock to 0 or leaves it unchanged.

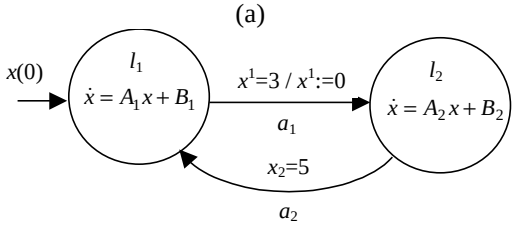
If in the above definition $act_l : \dot{x} = k$, with $k \in \mathbf{Z}$ an integer nonzero constant, and for all transitions having the source in l, $a = (l, l')$ with $l' \in Loc$, the assignment is defined by: Ass_a: $x := g(x) \in \{0, x\}$, then x is a *skewed clock*.

2.2 Examples and comments

Example 1 : the level controller (Favela, 1999)



(a)



(b)

Fig. 1. Hybrid system representing the controlled level dynamics in a water tank (a); hybrid automaton (b).

Consider the hybrid automaton in fig. 1(b). The level x^2 in the water tank in fig. 1(a) is controlled by an automaton that measures it and opens or closes the valve S. The state variable x^1 represents the clock of the controller ($\dot{x}^1 = 0$). When the controller closes the valve (location l_1 in the hybrid automaton), then

the level rises with the speed $\dot{x}^2 = -0.5x^2 + 5.15$. When the valve is open (location l2 in fig. 1(b)), then the level goes down with the speed $\dot{x}^2 = -0.4x^2 + 1$. In each location of the hybrid automaton, the evolution laws are represented by linear state equations, with the matrices:

$$A_1 = \begin{bmatrix} 0 & 0 \\ 0 & -0.5 \end{bmatrix},\ B_1 = \begin{bmatrix} 1 \\ 5.15 \end{bmatrix},$$
$$A_2 = \begin{bmatrix} 0 & 0 \\ 0 & -0.4 \end{bmatrix},\ B_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

The hybrid system has the following evolution. Suppose that $x(0) = [1.33 \quad 5]^T$ and the valve S is closed (location l_1). When the clock x^1 reaches the value 3, then the controller opens the valve S (the transition a_1 is fired) and resets the clock x^1 to zero. In the location l_2, when the water level x^1 reaches the value 5, then the controller closes the valve S (transition a_2 is fired) and the hybrid automaton returns to location l_1.

□

Example 2 (Alur *et al*., 1995). This example presents a timing based algorithm that implements a mutual exclusion protocol for a distributed system with *skewed clocks*. Consider the asynchronous shared-memory system that consists of two processes P_1 and P_2 with atomic read and write operations. Each process has a critical section and at each time instant, at most one of two processes is allowed to be in its critical section. Mutual exclusion is ensured by the following protocol described in pseudocode.

For each process P_i, where i = 1,2:

```
repeat
        repeat
                await k=0
                k:=i
                delay b
        until k=i
                Critical section
                k:=0
forever
```

The two processes share the variable k and process P_i is allowed to be in its critical section iff $k=i$. Each process has a private clock. The instruction **delay** b delays a process with at least b time units measured by the process's local clock. Furthermore, each process takes at most a time units, as measured by the process's local clock, for a single write access to the shared memory (i.e. for the assignment $k:=i$). The values a and b are the only information about the timing behavior of the instructions. Clearly, the protocol ensures mutual exclusion only for certain values of a and b. If both private processors locks are at precisely at the same rate, them mutual exclusion is guaranteed if $a < b$.

To make the example more interesting, in (Alur *et al*., 1995) is assumed that the two private clocks of the processes P_1 and P_2 proceed at different rates, namely the local clock of P_2 is 1.1 times faster than the clock of P_1. The resulting system can be modeled by the product of the two hybrid systems presented in fig. 2. Each of the two graph models one process, with the two critical sections being represented by the locations 4 and D. the private clocks of the processes $P_1$and P_2 determine the rate of change of the two skewed-clock variables x and y respectively.

□

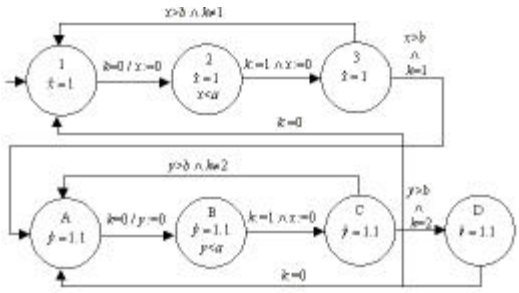


Fig. 3. Mutual exclusion protocol.

3. COMMENTS AND PROBLEMS REGARDING CLOCK MODELS IN HYBRID CONTROL SYSTEMS WITH INTERFACE

3.1 Hybrid control systems with interface – a brief review

The hybrid control system with interface consists of a continuous plant that is controlled, through an interface, by a DES controller. Starting from a primal control objective, modeled as a string of inequality restrictions imposed to the continuous state, the continuous state space is partitioned, by smooth hypersurfaces, into a finite number of cells. A plant event is generated whenever the continuous trajectory crosses, in a certain direction, a hypersurface of the partition. The continuous plant receives a piecewise constant control signal and evolves from cell to cell, generating plant-events for the controller. The controller receives a plant-event and instantly fires an internal state transition, generating a control-symbol for the plant. The current control-symbol produces, through the interface, a switch of the control signal to a specific value and so on.

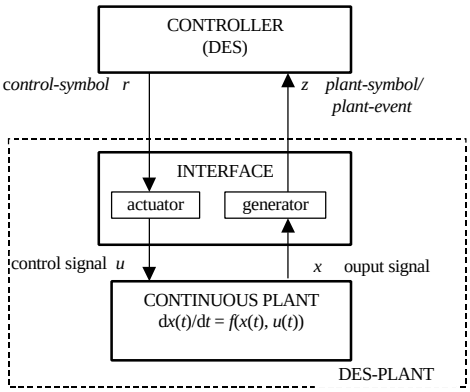


Fig.4. The structure of the HCS with interface.

The continuous plant is modelled by the differential system

$$\dot{x}(t) = f(x(t), u(t))$$

where $x(t) \in X \subseteq \mathbf{R}^n$ and $u(t) \in U \subset \mathbf{R}^{mc}$ are the state and control vector respectively, at the time $t \in \Im = \mathbf{R}$. X is the continuous state space. The set of admissible control values $U = \{u_1, \ldots, u_M\}$ is bijectively related to the *alphabet of control-symbols* $R = \{r_1, \ldots, r_M\}$.

The nonsingular hypersurfaces $\mathrm{Ker}(h_i)=\{x \in X \mid h_i(x) = 0,\ h_i : X \to \mathbf{R}$ smooth $\}$, $i = 1, \ldots, N$, define a partition of $Q \le 2^N$ disjoint open cells, labeled by the alphabet of the plant's discrete states $P = \{p_1, \ldots, p_Q\}$. The alphabet of plant-symbols is $Z = \{z_{1+}, z_{1-}, \ldots, z_{N+}, z_{N-}\}$ and it consists of the labels associated to the possible plant events.

Definition 2. The DES-plant models the plant coupled to the interface and is defined by a nondeterministic automaton $G_p = \{ P, R, f_p, Z, g_p \}$, where P is the set of discrete states, R is the input alphabet of control-symbols, Z is the output alphabet of plant-symbols, $f_p : P \times R \to 2^P$ is the state transition function and $g_p : P \times P \to Z$ is the output function. The dynamic equations are

$$p(k+1) \in f_p(p(k), r(k)),\ g_p(p(k), p(k+1)) = z(k+1),$$

where $p(k) \in P$, $z(k+1) \in Z$, $r(k) \in R$ and $p(k) \neq p(k+1)$, $\forall\, k \in \{0, 1, \ldots\}$. □

G_p is the logical model of all possible behaviors of the plant, governed by all possible control signals generated by sequences of control-symbols from R and with initial conditions located, arbitrarily, in some initial cell of the partition.

Definition 2. The DES controller is a deterministic Moore machine $G_c = \{ S, Z, f_c, s_0, R, g_c \}$, where S is the finite set of discrete states, $s_0 \in S$ is the initial state, Z is the input alphabet, R is the output alphabet, $f_c : S \times Z \to S$ is the state transition function and $g_c : S \to R$ is the output function. The dynamic equations are

$$f_c(s(k), z(k+1)) = s(k+1),\ g_c(s(k+1)) = r(k+1),$$
$$s(0) = s_0,\ g_c(s(0)) = r(0).$$

□

3.2 Comments on a rule-based implementation of the DES controller and an associated discrete clock

Denote $stctr(k)$, $inctr(k)$ and $ouctr(k)$ the state, input and output of the rule-based system at the discrete time moment k, respectively. A possible rule-based implementation of the DES controller G_c is the following:

Rc1. if $stctr(k) = s_0$ then $ouctr(k) = g_c(0)$
NOTE: initialization
For any $k \ge 0$:
Rc2. if $stctr(k) = s(k)$ and $inctr(k+1) = z(k+1)$
then $stctr(k+1) = s(k+1)$ and
$ouctr(k+1) = r(k+1)$.
NOTE: the transition $s(k) \to s(k+1)$.

A HORN structure implies a premise, with multiple terms, related by logical connectives (AND, OR) and a conclusion represented by a single term (Cârstoiu, 1994). Because the right hand side of the rule Rc2 contains two terms and the next state and output are generated at the same discrete moment, Rc2 cannot be implemented as a HORN clause. This means that in real software implementations, there is a delay between the generation of the next state and of the next output of the controller, respectively, but, if the control part is much quickly than the plant, this delay may not be taken into account.

Note that in the above rule-based implementation, the discrete variable k is the asynchronous clock, respectively associated to the events that occur in the plant – i.e. to the crossing of a hypersurfaces, in the partitioned state space X, by the continuous trajectory $x(.)$. The rule Rc2 can be modified into RC2', by means of an additional internal state *var*, in order to reflect a HORN structure. However, the price paid is the introduction of an additional rule Rc2", which refers the dynamics of the supplementary internal clock k_1, in order to reflect the state and output transitions.

Rc2'. if $stctr(k) = s(k)$ and $inctr(k+1) = z(k+1)$
then $var(k) = var(k+1)$.
Rc2". if $var(k+1)$ then $stctr(k+1) = s(k+1)$ and
$k_1 = k_1+1$ and $ouctr(k+1) = r(k+1)$ and $k_1 = k_1+1$.

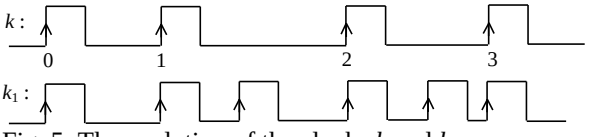


Fig. 5. The evolution of the clocks k and k_1.

When $k = 0$, the internal clock k_1 is set to the value of k. Hence, the discrete variable k is global and it

controls the closed loop representing the hybrid control system, while the additional internal clock k_1 is local and it times the state and output transitions of the DES controller, which cannot occur at the same time instant. The implicit assumption is that no additional plant event occurs while the controller changes its current state and output.

3.3 Comments on an extension of the HCS approach to temporal tasks and related clock models – an example

A possible extension of the HCS structure to timed control tasks is illustrated by means of a simple example, which is similar to the one described in Fig. 1(a). We shall consider first a pure logic control task, and the associated HCS will be intuitively built. The logic control task is then enriched with a temporal restriction, and the corresponding hybrid model of the closed loop hybrid system is also intuitively extracted.

The HCS associated to a pure logic control task. Consider the generic filling process depicted in fig. 6(a). The dynamics of the water level x is described by the state equation

$$\dot{x} = u_1 - u_2 ,$$

and the two valves, controlled by the bipositional signals u_1 and respectively u_2, cannot simultaneously be open (i.e. the combination $u_1 = 1$, $u_2 = 1$ is forbidden). The control-symbols are defined, respectively, by the switchings:

$r_1 \Rightarrow (u_1 \rightarrow 1, u_2 \rightarrow 0)$,
$r_2 \Rightarrow (u_1 \rightarrow 0, u_2 \rightarrow 1)$,
$r_3 \Rightarrow (u_1 \rightarrow 0, u_2 \rightarrow 0)$,

and the alphabet of control-symbols is

$$R = \{r_1, r_2, r_3\}.$$

Given the above defined continuous dynamics and the alphabet of control-symbols, *the logic control task* is to drive the water level according to the cyclic sequence

$$S := c_1, c_2, c_1, c_2, \ldots,$$

where the cells c_1 and c_2 are defined as follows:

$$c_1: 0 < x < \mathrm{L1},\ c_2: \mathrm{L1} < x < \mathrm{L2}.$$

This suggests the placement of the threshold sensors, denoted also L1 and L2, respectively, which define the alphabet of plant-events

$$Z = \{\mathrm{L1+}, \mathrm{L1-}, \mathrm{L2+}, \mathrm{L2-}\} \cup \{\varepsilon\}.$$

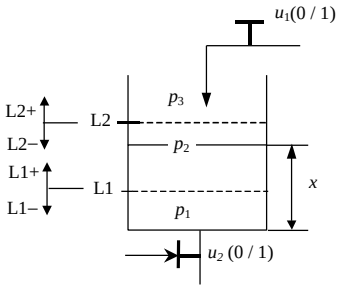
For instance, L1+ is the symbol associated to the plant-event described by the relations $dx(t) / dt > 0$ and $x(t) = \mathrm{L2}$. ε labels the silent plant-event, which has no effect for the DES-controller. The threshold sensors define also the state space partition depicted in fig. 6(a), formally described by the alphabet of discrete states

$$P = \{p_1, p_2, p_3\}$$

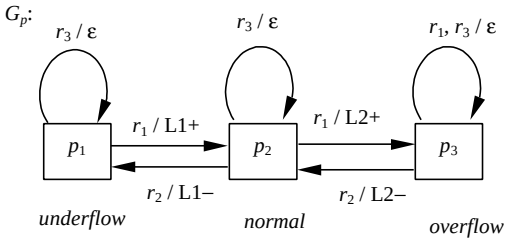
of the DES-plant model (fig.6(b)). Taking into account that the continuous plant is a pure integrator, the DES-plant model in fig. 6(b) is intuitively extracted. The control objective, or *the logic control task*, is represented, in the DES-plant model G_p, by the discrete evolution

$$w_p = p(0),p(1),p(2),p(3), \ldots = p_1p_2p_1p_2 \ldots .$$

A corresponding DES-controller is the Moore automaton G_c, depicted in fig. 7(a).

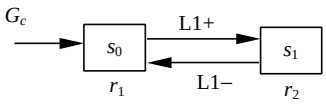


(a)

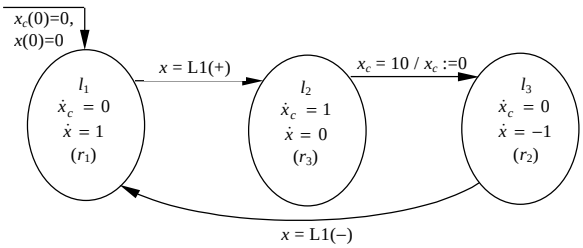


(b)

Fig. 6. The water tank with the threshold sensors (a) and the associated DES-plant model (b).



(a)



(b)

Fig. 7. The pure logic DES controller, modeled as a Moore machine (a) and the hybrid automaton representing the closed loop evolution of the water tank coupled to the temporal controller (b).

The hybrid closed loop model associated to a logic and temporal control task. Assume that the above described logic control task is transformed as follows: starting from p_1, the water level has to reach p_2, where it has to remain 10 time units and then it has to go back to p_1, and so on, i.e. the desired evolution can be described by the sequence

$$S_{\text{temp}} := p_1,p_2,wait(10),p_1,p_2,wait(10),\dots .$$

This leads to the necessity to include, at the control level, a *clock* described by the evolution of the continuous variable x_c. A hybrid automaton with three locations, as shown in fig. 7(b), now describes the controlled closed loop system. In the first location, l_1, the control value $u = u_1 - u_2$ corresponds to the one generated by the former control-symbol r_1, in the second location l_2 the control value u is null and in the third one the control value becomes –1, thus corresponding to the former control-symbol r_2.

It's obvious that the controller cannot be modeled any more as a Moore automaton, as depicted in fig.4 and in fig. 7(a) and the discrete transitions of the closed loop system depend on the occurrence of plant-events, but also on temporal conditions, which have a dynamic that is intrinsic to the controller.

4. CONCLUSIONS

This contribution has emphasized some specific aspects concerning importance of clocks in hybrid automata and in a class of hybrid control systems with interface.

Regarding the second class of hybrid models, two problems have been introduced: (*i*) starting from the necessity to implement the logic DES controller by means of dedicated rules having a HORN structure, a local discrete clock variable is introduced, with a dynamics that depend on the global variable that governs the closed loop timing; (*ii*) starting from the necessity to enrich a pure logic control task with temporal restrictions, it was shown, within a simple example, that a clock model has to be introduced at the control level and also that the entire closed loop hybrid system can be modeled as a hybrid automaton. In the second problem, it is obvious that, when introducing a temporal restriction, a Moore machine cannot model the DES controller anymore.

REFERENCES

Allam M. and H. Alla (1998). Performance evaluation of hybrid systems. In: *Proc. of the 3rd Int. Conf. on Automation of Mixed Proceses*, 19-20 March, Reims, France, 9-16

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. In: *Theoretical Comp. Science*, **138**, 3-34, Elsevier.

Cârstoiu D.(1994). *Expert Systems* (in Romanian), All Press.

Cârstoiu D. and V.E.Oltean (2001). Some Aspects Concerning a Rule-Based Representation of Hybrid Control and Supervision Systems. In: *Proc. of the 13th Int. Conf. on Control Systems and Computer Science CSCS13*, May 31 – June 2, Bucharest, Romania, 236-241

Favela Contreras A.R.X. (1999). *Modélisation et analyse du comportement dynamique des systèmes hybrides: une approche basée sur le modèle automate hybride*. Thèse pour obtenir le grade de docteur de L'Institut National Polytechnique de Grenoble.

Ramadge J.G. and W.M. Wonham (1989). The Control of Discrete Event Systems. In: *IEEE Proc*., **77(1)**, 81-98

Stiver J.A., P.J. Antsaklis and M.D. Lemmon (1994). *A Logical DES Approach to the Design of Hybrid Control Systems*. Technical Report of the ISIS Group at the University of Notre Dame, ISIS-94-011.