J. A. Stiver and P. J. Antsaklis, "On the Controllability of Hybrid Control Systems," P roc 3 2nd I EEE C o nference o n D ecision a nd C ontrol , pp. 294-299, San Antonio, TX, Dec. 15-17, 1993.

Proceedings of the 32nd Conference on Decision and Control
San Antonio, Texas • December 1993

On the Controllability of Hybrid Control Systems

James A. Stiver and Panos J. Antsaklis
Department of Electrical Engineering
University of Notre Dame, Notre Dame, IN 46556

Abstract

In this paper, it is shown how existing discrete event system (DES) methods can be used to design controllers for hybrid control systems. Specifically, we concentrate on the notions of controllability and of the supremal controllable sublanguage, developed by Ramadge and Wonham, and extend those notions to analyze hybrid control control systems using models developed by the authors. Using this notion of controllability, we are able to extend DES controller design methods and to design controllers for hybrid control systems.

1 Introduction

A hybrid control system is a system which contains both continuous and discrete dynamics. The continuous dynamics are modeled with a set of ordinary differential equations, while the discrete dynamics form a discrete event system (DES) which can be modeled with an automaton. A simple example of a hybrid control system is the heating and cooling system of a typical home. The furnace and air conditioner, along with the heat flow characteristics of the home, form a continuous-time system which is to be controlled. The thermostat is a simple discrete event system which basically handles the symbols *too hot, too cold,* and *ok*. The temperature of the room is translated into these representations in the thermostat and the thermostat's response is translated back to electrical currents which control the furnace, air conditioner, blower, etc. Additional applications of hybrid control systems include flexible manufacturing systems and chemical process controls, among others.

Recently, efforts have been made to study hybrid systems in a unified, analytical way, [1-11]. In [1], hybrid systems are modeled using a language called SIGNAL in which discrete-time signals and events are synchronized to the fastest clock. In [2] a modeling strategy for hybrid systems is suggested and used to model a computer hard drive. [3] describes a model for a class of systems containing both continuous and discrete dynamics and shows that this class of models display causality and time monotonicity, which are essential for these models to be used in on-line monitoring. In [6] a technique is discussed which learns a set of control policies, used in the hybrid system, to control the continuous-time plant.

In [4] and [5], the development of a formal model for hybrid systems is presented. The model provides a framework to represent the interaction between the continuous and discrete portions of the system. [8] presents a hybrid system model in which the discrete dynamics are modeled by a petri net and special attention is given to describing events and event structures. Finally, [9-11] contain background as well as some of the results reported in this work.

The hybrid control systems considered here consist of three distinct levels; see Figure 1. The controller is a discrete-state system, a sequential machine, seen as a Discrete Event System (DES). The controller receives, manipulates and outputs events represented by symbols. The plant is a continuous-state system typically modeled by differential/difference equations and it is the system to be controlled by the discrete-state controller. The plant receives, manipulates and outputs signals represented by real variables that are typically (piecewise) continuous. The controller and the plant communicate via the interface that translates plant outputs into events for the controller to use, and controller output events into command signals for the plant input. The interface consists of two subsystems: the event generator that senses the plant outputs and generates symbols representing plant events, and the actuator which translates the controller's symbolic commands into piecewise constant plant input signals.

It is possible to model a hybrid control system as a pair of interacting discrete event systems, where one of the DES's is the controller and the other represents the combined plant and interface. Modeling the system in this way allows the techniques developed for the analysis and design of DES controllers to be applied to hybrid systems. The DES techniques must be adapted, however, because the DES models used to represent hybrid control systems are not the same as those typically used in the study of logical discrete event systems.

It should be pointed out that the DES plant model used in our approach significantly facilitates the derivation of analytical results. This is mainly due to our choice for the interface; furthermore this was done without loss of generality. Comparable results have not been reported using alternative existing hybrid control system models. Note that a systematic, general methodology to design controllers for hybrid control systems has not appeared in the literature before to our knowledge. It is presented for the first time in this paper.

This paper describes the hybrid control system model and associated DES models and then presents extensions

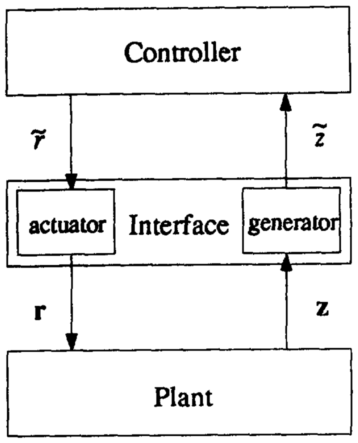

Figure 1: Hybrid Control System

to some of the DES tools developed by Ramadge and Wonham which allow these tools to be applied to hybrid control systems. In particular, a new result concerning the important concept of controllability is introduced, thus opening the way for the application of controller design methods. Two examples are included to illustrate these ideas. The design of hybrid control system controllers is also shown.

2 Hybrid Control System Model

A hybrid control system, can be divided into three parts as shown in Figure 1. The models we use for each of these three parts, as well as the way they interact are now described.

2.1 Plant

The system to be controlled, called the plant, is modeled as a time-invariant, continuous-time system. This part of the hybrid control system contains the entire continuous-time portion of the system, possibly including a continuous-time controller. Mathematically, the plant is represented by the familiar equations

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{r}) \quad (1)$$
$$\mathbf{z} = g(\mathbf{x}) \quad (2)$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{r} \in \mathbb{R}^m$, and $\mathbf{z} \in \mathbb{R}^p$ are the state, input, and output vectors respectively. $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^p$ are functions. For the purposes of this work we assume that $\mathbf{z} = \mathbf{x}$. Note that the plant input and output are continuous-time vector valued signals. Bold face letters are used to denote vectors and vector valued signals.

2.2 Controller

The controller is a discrete event system which is modeled as a deterministic automaton. This automaton can be specified by a quintuple, $\{\tilde{S}, \tilde{Z}, \tilde{R}, \delta, \phi\}$, where $\tilde{S}$ is the (possibly infinite) set of states, $\tilde{Z}$ is the set of *plant symbols*, $\tilde{R}$ is the set of *controller symbols*, $\delta : \tilde{S} \times \tilde{Z} \to \tilde{S}$ is the state transition function, and $\phi : \tilde{S} \to \tilde{R}$ is the output function. The symbols in set $\tilde{R}$ are called controller symbols because they are generated by the controller. Likewise, the symbols in set $\tilde{Z}$ are called plant symbols and are generated by the occurrence of events in the plant. The action of the controller can be described by the equations

$$\tilde{s}[n] = \delta(\tilde{s}[n-1], \tilde{z}[n]) \quad (3)$$
$$\tilde{r}[n] = \phi(\tilde{s}[n]) \quad (4)$$

where $\tilde{s}[n] \in S, \tilde{z}[n] \in \tilde{Z}$, and $\tilde{r}[n] \in \tilde{R}$. The index n specifies the order of the symbols in a sequence. The input and output signals associated with the controller are asynchronous sequences of symbols, rather than continuous-time signals. Notice that there is no delay in the controller. The state transition, from $\tilde{s}[n-1]$ to $\tilde{s}[n]$, and the controller symbol, $\tilde{r}[n]$, occur immediately when the plant symbol $\tilde{z}[n]$ occurs.

Tildes are used to indicate that the particular set or signal is made up of symbols. For example, $\tilde{Z}$ is the set of plant symbols and $\tilde{z}$ is a sequence of plant symbols. An argument in brackets, e.g. $\tilde{z}[n]$, represents the nth symbol in the sequence $\tilde{z}$. A subscript, e.g. $\tilde{z}_i$, is used to denote a particular symbol from a set.

2.3 Interface

The controller and plant cannot communicate directly in a hybrid control system because each utilizes a different type of signal. Thus an interface is required which can convert continuous-time signals to sequences of symbols and vice versa. The interface consists of two rather simple maps, α and γ.

The first map, called the *generator*, $\alpha : \mathbb{R}^n \times \mathbb{R}^n \to \tilde{Z}$, generates a sequence of plant symbols based upon the state of the plant. The generator, α, is based on a set of functions, $h_i(\mathbf{x})$, each of which forms a boundary in the state space. Together, these boundaries partition the state space into a number of regions. Whenever the state of the plant crosses a boundary, a *plant event* has occurred. Associated with each plant event is a plant symbol which is generated by the generator when the plant event is detected. Thus, the generator detects boundary crossings and reports each crossing via a plant symbol.

For a mathematical treatment of the generator we start by defining the sequence $\tau[n]$ which gives the times at which the plant events occur.

$$\tau[0] = 0 \quad (5)$$
$$\tau[n] = \inf\{t > \tau[n-1] : \exists i, h_i(\mathbf{x}(t)) \cdot h_i(\mathbf{x}(\tau[n-1] + \epsilon)) < 0\} \quad (6)$$

With the plant event times defined, the sequence of plant symbols can be defined as follows.

$$\tilde{z}[n] = \begin{cases} \tilde{z}_i^+ & \text{if} \quad h_i(\mathbf{x}(\tau[n] + \epsilon)) > 0 \\ \tilde{z}_i^- & \text{if} \quad h_i(\mathbf{x}(\tau[n] + \epsilon)) < 0 \end{cases} \quad (7)$$

where i is from equation 6 and ϵ is an infinitesimal positive real number. So we see that the set of plant symbols consists of two symbols for each boundary, which indicate which boundary has been crossed and in which direction.

The second map, called the *actuator*, $\gamma : \tilde{R} \rightarrow \mathbb{R}^m$, converts a sequence of controller symbols to a piecewise constant plant input as follows

$$\mathbf{r}(t) = \gamma(\tilde{r}[n]) \tag{8}$$

where

$$n = \max\{m : \tau[m] < t\} \tag{9}$$

and $\tau[m]$ is the sequence defined in equation 6. The plant input, $\mathbf{r}(t)$, can only take on certain specific values, where each value is associated with a particular controller symbol. Thus the plant input is a piecewise constant signal which changes only when controller symbols occur.

Part of this event characterization was suggested by M. Lemmon. In earlier formulations [10,11] events were associated with regions rather than boundaries.

2.4 DES Plant Model

If the plant and interface of a hybrid control system are viewed as a single component, this component behaves like a discrete event system. It is advantageous to view a hybrid control system this way because it allows it to be modeled as two interacting discrete event systems which are more easily analyzed than the system in its original form. The discrete event system which models the plant and interface is called the *DES Plant Model* and is modeled as an automaton similar to the controller. The automaton is specified by a quintuple, $\{\tilde{P}, \tilde{Z}, \tilde{R}, \psi, \xi\}$, where $\tilde{P}$ is the set of states, $\tilde{Z}$ and $\tilde{R}$ are the sets of plant symbols and controller symbols, $\psi : \tilde{P} \times \tilde{Z} \rightarrow \tilde{P}$ is the state transition function, and $\xi : \tilde{P} \times \tilde{R} \rightarrow \mathrm{P}(\tilde{Z})$ is the enabling function.

The behavior of the DES plant model is as follows

$$\begin{aligned} \tilde{z}[n+1] &\in \xi(\tilde{p}[n], \tilde{r}[n]) & (10) \\ \tilde{p}[n] &= \psi(\tilde{p}[n-1], \tilde{z}[n]) & (11) \end{aligned}$$

where $\tilde{p}[n] \in \tilde{P}, \tilde{r}[n] \in \tilde{R}$, and $\tilde{z}[n] \in \tilde{Z}$. The enabling function defines which events are enabled in a given state and input. The state transition function defines the state which results following the occurrence of an event. The state transition function, ψ, is a partial function because some events are never enabled from a given state. In general, more than one event will be enabled for a given state and input, and therefore the DES plant may be nondeterministic.

If we let $\tilde{Z}^*$ be the set of all strings of plant symbols, then the state transition function can also be used to generate the plant state after a string of events.

$$\tilde{p}[n] = \psi(\tilde{p}[0], \tilde{w}) \tag{12}$$

where $\tilde{w} \in \tilde{Z}^*$ of length n.

The set of states, $\tilde{P}$, of the DES plant model is based upon the partition realized in the generator. Specifically, each state in $\tilde{P}$ corresponds to a region, in the state space of the continuous-time plant.

As mentioned above, the DES plant model will generally be nondeterministic which can be a problem from the standpoint of control. To avoid this problem the boundaries which define the plant events must be chosen in a way which yields a manageable DES plant model. This is not the topic of this paper, however a treatment of this issue can be found in [12] under quasideterminism.

3 DES Models for Hybrid Control Systems

In the previous section, a DES plant model was described which allows the plant and interface of a hybrid control system to be modeled as a DES. Together with the controller, the DES plant forms a DES model for the hybrid control system called the HDES. Before existing DES techniques developed in the Ramadge-Wonham framework can be extended, certain differences must be dealt with. The models used in the Ramadge-Wonham framework differ from the HDES and therefore the theorems and techniques cannot be applied directly, but must be adapted. In this section the differences will be outlined and explained.

The Ramadge-Wonham model (RWM) consists of two interacting DES's called the generator and supervisor. The generator is analogous to the DES plant model of a HDES and the supervisor is analogous to the controller. Using our notation, the generator of the RWM is an automaton consisting of a state set, $\tilde{P}$, a set of event labels or symbols, $\tilde{Z}$, and a state transition function, $\psi : \tilde{P} \times \tilde{Z} \rightarrow \tilde{P}$. So far this is the same as our DES plant model, the difference appears in the way events are enabled.

The events in the RWM are divided into two sets, those which are controllable and those which are uncontrollable: $\tilde{Z} = \tilde{Z}_c \cup \tilde{Z}_u$. The events in $\tilde{Z}_c$ can be individually enabled or disabled by commands from the supervisor, while the events in $\tilde{Z}_u$ are always enabled. The supervisor can enable any subset of $\tilde{Z}_c$, and it's ability to enable and disable events is not changed by the state of the generator.

This is in contrast to our DES plant model where each command (controller symbol) from the DES controller enables a particular subset of $\tilde{Z}$ determined by ξ, and this subset depends not only on the controller symbol but also the present state of the DES plant model. In addition, there is no guarantee that any arbitrary subset of $\tilde{Z}$ can be enabled while other plant events are not enabled. The general inability to enable events individually and the enabling function's dependence upon the state of the DES plant model, are what differentiate the HDES from the RWM DES of the Ramadge-Wonham framework.

The reason for the differences between the RWM generator and the DES plant model is that the DES plant model is not representing a true discrete event system but

J. A. Stiver and P. J. Antsaklis, "On the Controllability of Hybrid Control Systems," P roc 3 2nd I EEE C o nference o n D ecision a nd C ontrol , pp. 294-299, San Antonio, TX, Dec. 15-17, 1993.

rather a continuous-state plant with an interface. This means that each state in the DES plant model corresponds to a subset of states in the actual plant. Since continuous-state systems will react differently to inputs depending on their current state, the DES plant model will react differently to inputs depending on its current state. A consequence of the actual plant having a continuous rather than a discrete state space is that an event consists of a subset of the state space rather than one specific state transition. This generally prevents the individual enabling of particular events.

3.1 Controllability of the HDES

The behavior of a DES can be characterized by the set of possible event sequences which it can generate. This set is referred to as the language of the DES, denoted L, and defined as

$$L = \{w : w \in \tilde{Z}^*, \phi(p_0, w) \text{ is defined}\} \tag{13}$$

where $\tilde{p}_0 \in \tilde{P}$. This section outlines new results about the controllability of languages in a HDES.

When a DES is controlled by another system such as a RWM supervisor or a DES controller, the DES will generate event strings which lie in a subset of its language. If we denote the language of the DES under control as L_c then $L_c \subset L$. A theorem has been developed to determine whether a given RWM generator can be controlled to a desired language [13]. That is, whether it is possible to design a controller such that the DES will be restricted to some target language K. The theorem states that it is possible if

$$\bar{K}\tilde{Z}_u \cap L \subset \bar{K} \tag{14}$$

where $\bar{K}$ represents the set of all prefixes of K. When this condition is met for a generator of language L, the language K is said to be controllable, and a controller can be designed which will restrict the generator to the language K. This condition requires that if an uncontrollable event occurs after the generator has produced a prefix of K, the resulting string must still be a prefix of K because the uncontrollable event cannot be prevented.

For the HDES, the language K is controllable if

$$\forall w \in \bar{K} \; \exists \tilde{r} \in \tilde{R} \ni w\xi(\psi(\tilde{p}_0, w), \tilde{r}) \subset \bar{K} \tag{15}$$

This condition requires that for every prefix of the desired language, K, there exists a control, $\tilde{r}$, which will enable only events which will cause string to remain in K.

Theorem 1 *If K is prefix closed and controllable according to 15, then a DES controller can be designed which will control the DES plant model to the language K.*

Proof: The proof can be found in [14].

Since the DES plant model can be seen as a generalization of the RWM, the conditions in 15 should reduce to those of 14 under the appropriate restrictions. This is indeed the case.

If the desired behavior (i.e. language) is not attainable for a given controlled DES, it may be possible to find a more restricted behavior which is. If so, the least restricted behavior is desirable. [13] and [15] describe and provide a method for finding this behavior which is referred to as the *supremal controllable sublanguage*, $K^\uparrow$, of the desired language. The supremal controllable sublanguage is the largest subset of K which can be attained by a controller. $K^\uparrow$ can be found via the following iterative procedure.

$$\begin{aligned} K_0 &= K && (16) \\ K_{i+1} &= \{w : w \in \bar{K}, \bar{w}\tilde{Z}_u \cap L \subset \overline{K_i}\} && (17) \\ K^\uparrow &= \lim_{i \to \infty} K_i && (18) \end{aligned}$$

For hybrid control systems, the supremal controllable sublanguage of the DES plant model can be found by a similar iterative scheme.

$$\begin{aligned} K_0 &= K && (19) \\ K_{i+1} &= \{w : w \in \bar{K}, \forall v \in \overline{w} \; \exists \tilde{r} \in \tilde{R} \ni \\ & \quad v\xi(\psi(\tilde{p}_0, v), \tilde{r}) \in \overline{K_i})\} && (20) \\ K^\uparrow &= \lim_{i \to \infty} K_i && (21) \end{aligned}$$

Theorem 2 *For a DES plant model and language K, $K^\uparrow$ is controllable and contains all controllable sublanguages of K.*

Proof: The proof can be found in [14].

With this controllability result we are able to design controllers for hybrid control systems. The procedure is illustrated in the second example.

4 Examples

This section contains two examples which illustrate the material of this paper.

4.1 Example 1 - Double Integrator

Using the double integrator example from [10], we have the following plant

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{r} \tag{22}$$

In the interface, the function α partitions the state space into four regions as follows,

$$\begin{cases} \tilde{p}_1 & \text{if} \quad x_1, x_2 > 0 \\ \tilde{p}_2 & \text{if} \quad x_1 < 0, x_2 > 0 \\ \tilde{p}_3 & \text{if} \quad x_1, x_2 < 0 \\ \tilde{p}_4 & \text{if} \quad x_1 > 0, x_2 < 0 \end{cases}, \tag{23}$$

and the function γ provides three set points,

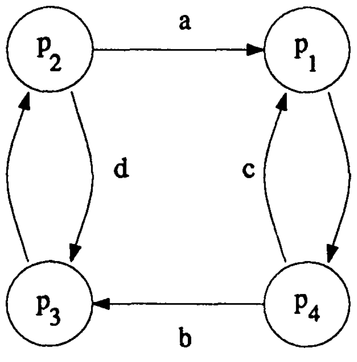

Figure 2: DES Plant Model for Example 1

$$\gamma(\tilde{r}) = \begin{cases} -10 & \text{if} \quad \tilde{r} = \tilde{r}_1 \\ 0 & \text{if} \quad \tilde{r} = \tilde{r}_2 \\ 10 & \text{if} \quad \tilde{r} = \tilde{r}_3 \end{cases}, \tag{24}$$

Combining the plant and interface, we obtain the DES plant model. This DES is represented by the automaton in figure 2, and explicitly by the following sets and functions.

$$\tilde{P} = \{\tilde{p}_1, \tilde{p}_2, \tilde{p}_3, \tilde{p}_4\} \tag{25}$$

$$\tilde{Z} = \{\tilde{a}, \tilde{b}, \tilde{c}, \tilde{d}\} \tag{26}$$

$$\tilde{R} = \{\tilde{r}_1, \tilde{r}_2, \tilde{r}_3\} \tag{27}$$

$$\psi(\tilde{p}_1, \tilde{d}) = \tilde{p}_4 \quad \psi(\tilde{p}_3, \tilde{c}) = \tilde{p}_2 \tag{28}$$

$$\psi(\tilde{p}_2, \tilde{a}) = \tilde{p}_1 \quad \psi(\tilde{p}_4, \tilde{c}) = \tilde{p}_1 \tag{29}$$

$$\psi(\tilde{p}_2, \tilde{d}) = \tilde{p}_3 \quad \psi(\tilde{p}_4, \tilde{b}) = \tilde{p}_3 \tag{30}$$

$$\xi(\tilde{p}_1, \tilde{r}_1) = \{\tilde{d}\} \quad \xi(\tilde{p}_3, \tilde{r}_3) = \{\tilde{c}\} \tag{31}$$

$$\xi(\tilde{p}_2, \tilde{r}_1) = \{\tilde{a}, \tilde{d}\} \quad \xi(\tilde{p}_4, \tilde{r}_1) = \{\tilde{b}\} \tag{32}$$

$$\xi(\tilde{p}_2, \tilde{r}_2) = \{\tilde{a}\} \quad \xi(\tilde{p}_4, \tilde{r}_2) = \{\tilde{b}\} \tag{33}$$

$$\xi(\tilde{p}_2, \tilde{r}_3) = \{\tilde{a}\} \quad \xi(\tilde{p}_4, \tilde{r}_3) = \{\tilde{b}, \tilde{c}\} \tag{34}$$

The values, for which ψ has not been stated, are undefined, and the values for which ξ has not been stated are the empty set.

The language generated by this automaton is $L = \overline{((dc)^* + db(cd)^*ca)^*}$. If we want to drive the plant in clockwise circles, then the desired language is $K = \overline{(dbca)^*}$. It can be shown that this K satisfies equation (15) and therefore according to theorem 1, a controller can be designed to achieve the stated control goal.

4.2 Example 2 - A More Complex DES Plant Model

This example has a richer behavior and will illustrate the generation of a supremal controllable sublanguage. We start immediately with the DES plant model shown in figure 3. The enabling function, ξ, is given by the following table.

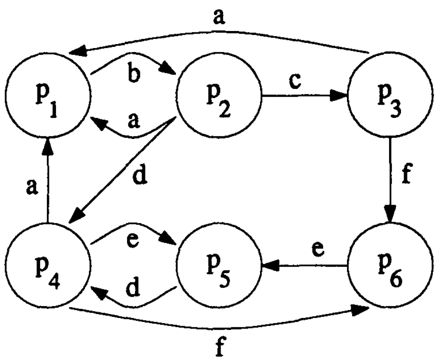

Figure 3: DES Plant Model for Example 2

ξ	$\tilde{r}_1$	$\tilde{r}_2$	$\tilde{r}_3$	$\tilde{r}_4$
$\tilde{p}_1$	$\emptyset$	$\{\tilde{b}\}$	$\{\tilde{b}\}$	$\emptyset$
$\tilde{p}_2$	$\{\tilde{a}\}$	$\{\tilde{a}, \tilde{d}\}$	$\{\tilde{c}\}$	$\{\tilde{a}, \tilde{c}, \tilde{d}\}$
$\tilde{p}_3$	$\{\tilde{a}\}$	$\{\tilde{f}\}$	$\emptyset$	$\{\tilde{a}, \tilde{f}\}$
$\tilde{p}_4$	$\{\tilde{a}\}$	$\{\tilde{f}\}$	$\{\tilde{a}, \tilde{f}\}$	$\{\tilde{a}, \tilde{e}, \tilde{d}\}$
$\tilde{p}_5$	$\emptyset$	$\{\tilde{d}\}$	$\{\tilde{d}\}$	$\{\tilde{d}\}$
$\tilde{p}_6$	$\{\tilde{e}\}$	$\{\tilde{e}\}$	$\{\tilde{e}\}$	$\{\tilde{e}\}$

(35)

The language generated by this DES is $L = \overline{L_m}$ where

$$L_m = (b(a + d\sigma^* a + c(a + fed\sigma^* a)))^* \tag{36}$$

and $\sigma = ((e + fe)d)$. Suppose we want to control the DES so that it never enters state $\tilde{p}_5$ and can always return to state $\tilde{p}_1$. The desired language is therefore

$$K = \overline{(a + b + c + d + f)^* a} \tag{37}$$

In this example, the language K is not controllable. This can be seen by considering the string $bcf \in K$, for which there exists no $\tilde{r} \in \tilde{R}$ which will prevent the DES from deviating from K by generating e and entering state $\tilde{p}_5$.

Since K is not controllable, we find the supremal controllable sublanguage of K as defined in equation 21. The supremal controllable sublanguage is

$$K^\dagger = K_1 = \overline{(a + b + c + d + f)^* a - (bcfed\sigma^* a)^*} \tag{38}$$

Obtaining a DES controller once the supremal controllable sublanguage has been found is straight forward. The controller is a DES whose language is given by $K^\dagger$ and the output of the controller in each state, $\phi(\tilde{s})$, is the controller symbol which enables only transitions which are found in the controller. The existence of such a controller symbol is guaranteed by the fact that $K^\dagger$ is controllable. For Example 2, the controller is shown in Figure 4 and it's output function, ϕ, is as follows:

$$\phi(\tilde{s}_1) = \tilde{r}_2 \quad \phi(\tilde{s}_2) = \tilde{r}_4 \tag{39}$$

$$\phi(\tilde{s}_3) = \tilde{r}_1 \quad \phi(\tilde{s}_4) = \tilde{r}_1 \tag{40}$$

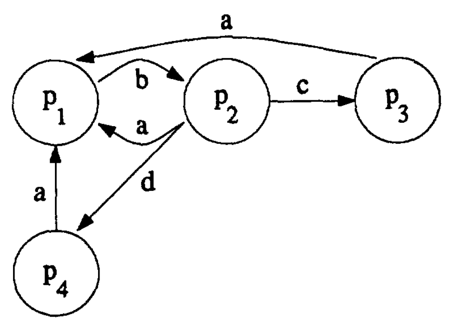

Figure 4: Controller for Example 2

5 Conclusion

We have extended the concepts of the controllability and the supremal controllable sublanguage of a given language, so that they may be applied to hybrid control systems. Various controller design techniques, developed for DES's, can now be applied to HCS's.

The relationship between the controllability of the continuous-time plant and the controllability of the DES plant model has not been fully examined. This issue needs further study. Nevertheless, it is not difficult to see that the two controllability concepts are related. If the continuous-time plant is not controllable, then it is likely that certain event sequences will not be achievable and this certainly affects the controllability of the DES plant model.

It should be noted that the problem of how properties of the continuous-time plant, such as controllability, observability, and stability, affect similar properties in DES plant model is currently under further investigation by the authors.

Acknowledgement - the partial support of NSF grant MSS-9216559 is acknowledged.

References

[1] A. Benveniste and P. Le Guernic, "Hybrid dynamical systems and the signal language", *IEEE Transactions on Automatic Control*, vol. 35, no. 5, pp. 535–546, May 1990.

[2] A. Gollu and P. Varaiya, "Hybrid dynamical systems", In *Proceedings of the 28th Conference on Decision and Control*, pp. 2708–2712, Tampa, FL, Dec. 1989.

[3] L. Holloway and B. Krogh, "Properties of behavioral models for a class of hybrid dynamical systems", In *Proceedings of the 31st Conference on Decision and Control*, pp. 3752–3757, Tucson, AZ, Dec. 1992.

[4] W. Kohn and A. Nerode, "Multiple agent autonomous hybrid control systems", In *Proceedings of the 31st Conference on Decision and Control*, pp. 2956–2966, Tucson, AZ, Dec. 1992.

[5] A. Nerode and W. Kohn, "Models for Hybrid Systems: Automata, Topologies, Stability", Private Communication, Nov. 1992.

[6] M. D. Lemmon, J. A. Stiver, and P. J. Antsaklis, "Learning to coordinate control policies of hybrid systems", In *Proceedings of the American Control Conference*, pp. 31–35, San Francisco, CA, June 1993.

[7] K. M. Passino and U. Ozguner, "Modeling and analysis of hybrid systems: Examples", In *Proceedings of the 1991 IEEE International Symposium on Intelligent Control*, pp. 251–256, Arlington, VA, Aug. 1991.

[8] P. Peleties and R. DeCarlo, "A modeling strategy with event structures for hybrid systems", In *Proceedings of the 28th Conference on Decision and Control*, pp. 1308–1313, Tampa, FL, Dec. 1989.

[9] J. A. Stiver and P. J. Antsaklis, "A novel discrete event system approach to modeling and analysis of hybrid control systems", In *Proceedings of the Twenty-Ninth Annual Allerton Conference on Communication, Control, and Computing*, University of Illinois at Urbana-Champaign, Oct. 1991.

[10] J. A. Stiver and P. J. Antsaklis, "Modeling and analysis of hybrid control systems", In *Proceedings of the 31st Conference on Decision and Control*, pp. 3748–3751, Tucson, AZ, Dec. 1992.

[11] J. A. Stiver and P. J. Antsaklis, "State space partitioning for hybrid control systems", In *Proceedings of the American Control Conference*, pp. 2303–2304, San Francisco, California, June 1993.

[12] P. J. Antsaklis, M. D. Lemmon, and J. A. Stiver, "Hybrid system modeling and event identification", Technical Report of the ISIS Group ISIS-93-002, University of Notre Dame, Notre Dame, IN, January 1993.

[13] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes", Systems Control Group Report 8515, University of Toronto, Toronto, Canada, Nov. 1985.

[14] P. J. Antsaklis, M. D. Lemmon, and J. A. Stiver, "Learning to be autonomous: Intelligent supervisory control", Technical Report of the ISIS Group ISIS-93-003, University of Notre Dame, Notre Dame, IN, April 1993.

[15] W. M. Wonham and P. J. Ramadge, "On the supremal controllable sublanguage of a given language", Systems Control Group Report 8312, University of Toronto, Toronto, Canada, Nov. 1983.

[16] P. Ramadge and W. M. Wonham, "The control of discrete event systems", *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–89, Jan. 1989.