P. J. Antsaklis, M. D. Lemmon and J. A. Stiver, "Hybrid System Modeling and Event Identification,"
P roc. o f t he T hirty-First A nnual A llerton C o nference o n C ommunication C ontrol a nd C omputing ,
pp 64-73, Univ. of Illinois at Urbana-Champaign, Sept. 29-Oct. 1, 1993.

Hybrid System Modeling and Event Identification

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

Abstract

Hybrid control systems contain two distinct types of systems, continuous state and discrete-state, that interact with each other. Their study is essential in designing sequential supervisory controllers for continuous- state systems, and it is central in designing control systems with high degree of autonomy.

After an introduction to intelligent autonomous control and its relation to hybrid control, models for the plant, controller, and interface are introduced. The interface contains memoryless mappings between the supervisor's symbolic domain and the plant's nonsymbolic state space. The simplicity and generality afforded by the assumed interface allows us to directly confront important system theoretic issues in the design of supervisory control systems; such as determinism and quasideterminism. It is possible to formulate sufficient conditions which guarantee that the equivalent plant DES satisfies specified deterministic or quasi-deterministic relationships. These conditions are linear inequalities which can be solved for using well known convex programming algorithms such as the ellipsoid method or interior point algorithms. These convex programming algorithms provide very efficient means of solving these problems. In addition, these algorithms can be interpreted as on-line learning algorithms (inductive inference), which means that interface design can be performed as part of an on-line identification of the events used by the supervisor in controlling the plant.

1 Introduction

In the design of controllers for complex dynamical systems, there are needs today that cannot be successfully addressed with the existing conventional control theory. Heuristic methods may be needed to tune the parameters of an adaptive control law. New control laws to perform novel control functions to meet new objectives should be designed while the system is in operation. Learning from past experience and planning control actions may be necessary. Failure detection and identification is needed. Such functions have been performed in the past by human operators. To increase the speed of response, to relieve the operators from mundane tasks, to protect them from hazards, a high degree of autonomy is desired. To achieve this autonomy, high level decision making techniques for reasoning under uncertainty must be utilized. These techniques, if used by humans, may be attributed to intelligence. Hence, one way to achieve high degree of autonomy is to utilize high level decision making techniques, intelligent methods, in the autonomous controller. In our view, higher autonomy is the objective, and intelligent controllers are one way to achieve it. The need for quantitative methods to model and analyze the dynamical behavior of such autonomous systems presents significant challenges

Proc of the 34th Allerton Conference,
Sept 29-Oct 1, 1993.

well beyond current capabilities. It is clear that the development of autonomous controllers requires significant interdisciplinary research effort as it integrates concepts and methods from areas such as Control, Identification, Estimation, Communication Theory, Computer Science, Artificial Intelligence, and Operations Research. For more information on intelligent control see [1–12]. Note that it is also possible to attain higher degrees of autonomy by using methods that are not considered intelligent. It appears however that to achieve the highest degrees of autonomy, intelligent methods are necessary indeed.

1.1 Hierarchical Architectures

Typically, intelligent controllers involve hierarchical functional architectures. A description of a hierarchical functional architecture of a controller that is used to attain high degrees of autonomy in future space vehicles can be found in [2]. Hierarchies offer very convenient ways to describe the operation of complex systems and deal with computational complexity issues, and they are used extensively in the modeling of intelligent autonomous control systems. Such a hierarchical approach is taken here (and in [2] and [4]) to study intelligent autonomous and hybrid control systems. Typically hierarchies for intelligent systems consist of three levels, the Execution Level, the Coordination Level, and the Management and Organization Level [8–11].

Hybrid control systems do appear in the intelligent autonomous control system framework whenever one considers the Execution level together with control functions performed in the higher Coordination and Management levels. Examples include expert systems supervising and tuning conventional controller parameters, planning systems setting the set points of local control regulators, sequential controllers deciding which from a number of conventional controllers is to be used to control a system, to mention but a few. One obtains a hybrid control system of interest whenever one considers controlling a continuous-state plant (in the Execution level) by a control algorithm that manipulates symbols, that is by a discrete-state controller (in Coordination and/or Management levels).

In the remainder of this paper, a Discrete Event System (DES) model for hybrid control systems is developed in Section 2. This modeling framework is used in Section 3 to develop supervisory controllers. In Section 4, event identification using convex programming methods is discussed.

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. For a more detailed description and discussion see [13].

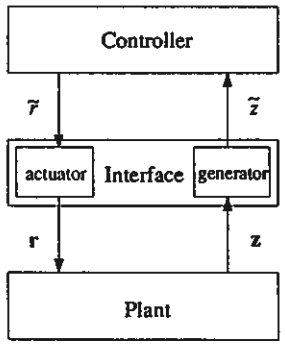

Figure 1: Hybrid Control System

P. J. Antsaklis, M. D. Lemmon and J. A. Stiver, "Hybrid System Modeling and Event Identification,"
P roc. o f t he T hirty-First A nnual A llerton C o nference o n C ommunication C ontrol a nd C omputing ,
pp 64-73, Univ. of Illinois at Urbana-Champaign, Sept. 29-Oct. 1, 1993.

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. The plant is represented by

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{r}) \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{r} \in \mathbb{R}^m$ are the state and input vectors respectively, and $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a function.

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 controller can be described by the equations

$$\tilde{s}[n] = \delta(\tilde{s}[n-1], \tilde{z}[n]) \tag{2}$$

$$\tilde{r}[n] = \phi(\tilde{s}[n]) \tag{3}$$

where $\tilde{s}[n] \in S, \tilde{z}[n] \in \tilde{Z}$, and $\tilde{r}[n] \in \tilde{R}$. 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.

2.3 Interface

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 *generator* is represented by a function, $\alpha : \mathbb{R}^n \to \tilde{X}$, and a set of functionals, $h_i : \mathbb{R}^n \to \tilde{X}$. The null space of each functional forms a separating hypersurface in the state space of the plant and whenever one of these surfaces is crossed a *plant event* occurs. With each plant event, the generator generates the following plant symbol.

$$\tilde{x}[n] = \alpha(\mathbf{x}(\tau[n])) \tag{4}$$

where $\tau[n]$ is the sequence of event instants or times defined as follows.

$$\tau[0] = 0 \tag{5}$$

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

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

$$\mathbf{r}(t) = \sum_{n=0}^{\infty} \gamma(\tilde{r}[n]) I(t) \tag{7}$$

where $I(t)$ is the following indicator function.

$$I(t) = \begin{cases} 1 & \text{if } \tau[n] \le t < \tau[n+1] \\ 0 & \text{if otherwise} \end{cases} \tag{8}$$

and $\tau[m]$ is the sequence defined in equation 6.

3 Controllability and DES Controller Design for Hybrid Control Systems

3.1 DES Models for Hybrid Control Systems

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*.

In this section we restrict the interface to systems where the plant symbols are associated with the separating hypersurfaces and the plant input is piecewise constant, which allows the DES to be modeled by an automaton. 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, $\psi : \tilde{P} \times \tilde{Z} \to \tilde{P}$ is the state transition function, and $\xi : \tilde{P} \times \tilde{R} \to \mathbf{P}(\tilde{Z})$ is the enabling function.

The behavior of the DES plant model is as follows

$$\tilde{z}[n+1] \in \xi(\tilde{p}[n], \tilde{r}[n]) \tag{9}$$

$$\tilde{p}[n] = \psi(\tilde{p}[n-1], \tilde{z}[n]) \tag{10}$$

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.

$$\tilde{p}[n] = \psi(\tilde{p}[0], \tilde{w}) \tag{11}$$

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

The models used in the Ramadge-Wonham framework differ from the above model, and therefore the theorems and techniques from that framework cannot be applied directly, but must be adapted. This is shown in [13] and will not be repeated here.

3.2 Controllability of the Hybrid Control System

For the hybrid control system, the language K is controllable if

$$\forall w \in \tilde{K}\ \exists\, \tilde{r} \in \tilde{R}\ \ni\ w\xi(\psi(\tilde{p}_0, w), \tilde{r}) \subset \tilde{K} \tag{12}$$

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 12, then a DES controller can be designed which will control the DES plant model to the language K.*

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

$$K_0 = K \tag{13}$$

$$K_{i+1} = \{w : w \in \tilde{K}, \forall v \in \overline{w}\ \exists\, \tilde{r} \in \tilde{R}\ \ni v\xi(\psi(\tilde{p}_0, v), \tilde{r}) \in \overline{K_i}\} \tag{14}$$

$$K^\dagger = \lim_{i \to \infty} K_i \tag{15}$$

P. J. Antsaklis, M. D. Lemmon and J. A. Stiver, "Hybrid System Modeling and Event Identification,"
P roc. o f t he T hirty-First A nnual A llerton C o nference o n C ommunication C ontrol a nd C omputing ,
pp 64-73, Univ. of Illinois at Urbana-Champaign, Sept. 29-Oct. 1, 1993.

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

With this controllability result we are able to design controllers for hybrid control systems. This is analogous to the controller design in the Ramadge-Wonham framework, and it is not described here.

4 Example - Double Integrator

Using the double integrator example from [14], we have the following plant

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{r} \tag{16}$$

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

$$\begin{cases} \tilde{p}_1 & \text{if} \quad x_1, x_2 > 0 \\ \tilde{p}_2 & \text{if} \quad x_1 < 0, x_2 > 0 \\ \tilde{p}_3 & \text{if} \quad x_1, x_2 < 0 \\ \tilde{p}_4 & \text{if} \quad x_1 > 0, x_2 < 0 \end{cases}, \tag{17}$$

and the function γ provides three set points,

$$\gamma(\tilde{r}) = \begin{cases} -10 & \text{if} \quad \tilde{r} = \tilde{r}_1 \\ 0 & \text{if} \quad \tilde{r} = \tilde{r}_2 \\ 10 & \text{if} \quad \tilde{r} = \tilde{r}_3 \end{cases}, \tag{18}$$

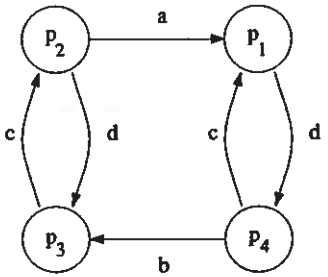

Figure 2: DES Plant Model for Example

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.

$$\tilde{P} = \{\tilde{p}_1, \tilde{p}_2, \tilde{p}_3, \tilde{p}_4\} \tag{19}$$
$$\tilde{Z} = \{\tilde{a}, \tilde{b}, \tilde{c}, \tilde{d}\} \tag{20}$$
$$\tilde{R} = \{\tilde{r}_1, \tilde{r}_2, \tilde{r}_3\} \tag{21}$$

$$\psi(\tilde{p}_1, \tilde{d}) = \tilde{p}_4 \quad \psi(\tilde{p}_3, \tilde{c}) = \tilde{p}_2 \tag{22}$$
$$\psi(\tilde{p}_2, \tilde{a}) = \tilde{p}_1 \quad \psi(\tilde{p}_4, \tilde{c}) = \tilde{p}_1 \tag{23}$$
$$\psi(\tilde{p}_2, \tilde{d}) = \tilde{p}_3 \quad \psi(\tilde{p}_4, \tilde{b}) = \tilde{p}_3 \tag{24}$$

$$\xi(\tilde{p}_1, \tilde{r}_1) = \{\tilde{d}\} \quad \xi(\tilde{p}_3, \tilde{r}_3) = \{\tilde{c}\} \tag{25}$$
$$\xi(\tilde{p}_2, \tilde{r}_1) = \{\tilde{a}, \tilde{d}\} \quad \xi(\tilde{p}_4, \tilde{r}_1) = \{\tilde{b}\} \tag{26}$$
$$\xi(\tilde{p}_2, \tilde{r}_2) = \{\tilde{a}\} \quad \xi(\tilde{p}_4, \tilde{r}_2) = \{\tilde{b}\} \tag{27}$$
$$\xi(\tilde{p}_2, \tilde{r}_3) = \{\tilde{a}\} \quad \xi(\tilde{p}_4, \tilde{r}_3) = \{\tilde{b}, \tilde{c}\} \tag{28}$$

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 (12) and therefore according to theorem 1, a controller can be designed to achieve the stated control goal.

5 Event Identification

The interface is instrumental in determining the *meaning* of the symbols used by the supervisor. Symbols acquire meaning by their association with nonsymbolic entities in the continuous-state plant. For example, a region of the state space or a control policy which successfully transfers a system between two regions of the state space may be used to ground symbols in a meaningful manner. In this section, a set of sufficient conditions are derived which test whether or not the control symbols used by the supervisor are meaningful with respect to their ability for supervising plant behaviour.

This test takes the form of a set of linear inequalities which can be efficiently solved using the method of centers [15]. The method of centers can also be framed as an inductive inference protocol so this method can be realized as an identification algorithm to be used for the on-line verification of a given DES plant automaton model. The significance of these results is that there are some very efficient method of centers algorithms. Prior work in this area has focused on the use of geometric centers such as in the ellipsoid algorithm [16]. While the ellipsoid algorithm is known to be somewhat inefficient, it provides provable results on computational complexity. These results suggest that the *event identification* scheme is learnable in the sense that the number of queries needed to decide on an event binding is bounded in a polynomial manner with plant complexity. Method of center algorithms based on analytic centers should provide faster learnability and the implementation issues associated with these algorithms are currently being investigated. The following two subsections first discuss the notion of supervisability used in this paper and then moves onto methods for solving the resulting inequality tests.

5.1 Supervisability

The combination of plant and interface forms another discrete event system. The hybrid control system, from the supervisor's perspective, can then be seen as two interconnected DES; the supervisor and an equivalent plant DES. Since the supervisor is directly connected to the plant DES, one approach to hybrid controller design is to synthesize a supervisor which effectively controls the equivalent DES model of the plant. The natural concern is whether or not control of the DES is sufficient to "acceptably" control the original plant.

The following discussion indicates precisely what is being controlled by the supervisor. Let $\mathbf{X}$ be a finite open cover of $\Re^n$ consisting of elements $\mathbf{x}_i$, for $i = 1, \ldots, q$. The set of *conjunctive events*, $\mathbf{C}$, generated by $\mathbf{X}$ will consist of all subsets, $\mathbf{c}$, of $\Re^n$ which can be expressed as the intersection of elements in $\mathbf{X}$. In other words, for any conjunctive event $\mathbf{c}$ there exists a collection, I, of integers between 1 and q such that

$$\mathbf{c} = \bigcap_{i \in I} \mathbf{x}_i \tag{29}$$

The set I will be called the index set of the conjunctive event.

P. J. Antsaklis, M. D. Lemmon and J. A. Stiver, "Hybrid System Modeling and Event Identification,"
P roc. o f t he T hirty-First A nnual A llerton C o nference o n C ommunication C ontrol a nd C omputing ,
pp 64-73, Univ. of Illinois at Urbana-Champaign, Sept. 29-Oct. 1, 1993.

The *symbolic* behaviour of the plant is described by the way in which the plant's state transitions between events. Consider a hybrid system with a collection, $\mathbf{C}$, of conjunctive events generated by a finite cover, $\mathbf{X}$, of the state space. Let $\tilde{R}$ be a finite alphabet of control symbols, let $V \subset \mathbf{C}$, and let $A \subset V \times V \times \tilde{R}$. The *plant automaton* associated with the hybrid system is a labeled directed graph, (V, A), where an ordered triple of $(\mathbf{c}_o, \mathbf{c}_t, \tilde{r})$ is in A if and only if the plant's state trajectory, $\bar{x}(t)$ generated by the system equations

$$\frac{d\bar{x}}{dt} = f(\bar{x}, \gamma(\tilde{r})) \tag{30}$$

satisfies the following conditions.

- there exists T_o such that $\bar{x}(t)$ is in the closure of $\mathbf{c}_o$ for $0 \leq t \leq T_o$.
- there exists T_t such that $\bar{x}(t)$ is in the closure of $\mathbf{c}_t$ for $t = T_t$.
- and $\bar{x}(t)$ is not in the closure of $\bigcup_i \mathbf{c}_i$ for $T_o < t < T_t$.

As a model of the plant's symbolic behaviour, the plant automaton provides the basis for determining a supervisor which can control the plant. In order for the resulting supervisor to effectively control the plant between conjunctive events, it is necessary that the plant automaton also be supervisable. The following makes this notion of supervisability more precise. Consider the plant automaton, (V, A) associated with a hybrid control system and consider two events, $\mathbf{c}_o$ and $\mathbf{c}_t$, which are elements of V. Let $\tilde{r}[n]$ denote the control history for a directed walk from vertex $\mathbf{c}_o$ to $\mathbf{c}_t$. The event $\mathbf{c}_t$ is said to be *supervisable from* $\mathbf{c}_o$ if and only if any other directed walk originating from $\mathbf{c}_o$ with control history $\tilde{r}[n]$ necessarily terminates at $\mathbf{c}_t$.

The following proposition provides a sufficient condition for testing whether or not an arc of a plant automaton is indeed supervisable

Proposition 1 *Consider a hybrid system with an associated basis event collection,* $\mathbf{X}$. *Let* (V, A) *be a given plant automaton. Let* $V_i : \Re^n \to \Re$ *(i=1,...,q) be a family of* C^1 *functionals and let* $f_j : \Re^n \to \Re^n$ *(*$j = 0, \ldots, m$*) be a family of Lipschitz continuous control policies. Let* $L_j V_i$ *be the Lie derivative of the functional* V_i *with respect to vector field* f_j. *The event* $\mathbf{c}$ *in* $\mathbf{C}$ *with index set* I *will be supervisable if*

- $V_i(\bar{x}) > 0$ *for all* $\bar{x} \notin \mathbf{c}_i$ *and* $V_i(\bar{x}) = 0$ *for all* $\bar{x} \in \mathbf{c}_i$.
- *for all* $i \in I$ *and for all* $\bar{x} \in \Re^n$,

$$0 > \begin{pmatrix} L_0 V_i & L_1 V_i & \cdots & L_m V_i \end{pmatrix} \begin{pmatrix} 1 \\ r_1 \\ \cdot \\ r_m \end{pmatrix} \tag{31}$$

- *and for all* $\bar{x}$ *on the boundary of events* $\mathbf{c}_o \neq \mathbf{c}$, *with index sets* I_o, *there exists an* $i \in I_o$ *such that* $V_i(\bar{x}) = 0$ *and*

$$0 < \begin{pmatrix} L_0 V_i & L_1 V_i & \cdots & L_m V_i \end{pmatrix} \begin{pmatrix} 1 \\ r_1 \\ \cdot \\ r_m \end{pmatrix} \tag{32}$$

Proof: (outline) In order for event $\mathbf{c}$ to be supervisable there must be an arc which goes from every event in V to $\mathbf{c}$ such that the state trajectory does not intersect any other event in V. One way to insure that this occurs is to require that $\mathbf{c}$ be a globally attracting invariant set

and that all other events are repellors of the plant controlled by $\gamma(\tilde{r})$. The above conditions are based in a straightforward manner on the LaSalle invariance principle. •

Remark: If the set of conjunctive events form a full partition of the plant's state space, then supervisability (determinism) is too restrictive to be useful. Insuring determinism over full state partitions will require that the partition boundaries be integral manifolds of the system; a condition which will probably be impossible to satisfy exactly in practice. The formulation stated above, however, does not require that events fully partition the state space. Therefore it is no longer necessary that event boundaries be integral manifolds. Consequently, there is significant flexibility in the way we can assign the control vectors $\bar{r}$ to insure determinism in the plant automaton.

Remark: The inequalities used to test the plant automaton's supervisability require that the Lie derivatives of the dynamic system be known. If there exist prior models for the plant, then these Lie derivatives can be estimated directly from state estimates. If no such models exist however, then these derivatives must be estimated by direct observation of the system. In this respect, the preceding inequalities indicate precisely what type of information must be made available to the system so that it can determine whether or not its plant automaton is indeed supervisable.

Remark: The conditions stated above are global statements. In many situations, however, satisfaction of these conditions on a local basis may be extremely useful. For example, if the plant's state trajectory is known to be contained within a given region of the state space, then it is only necessary that the above inequalities be satisfied in that region (rather than the entire state space) to insure the supervisability of the resulting plant automaton.

5.2 Identification

The sufficient conditions obtained in the preceding section form a system of inequalities which are linear in the control vector, $\bar{r}$. Since the control vector is determined by the hybrid system's actuator mapping, these conditions also provide a method for determining an actuator mapping, γ which insures that the given automaton will be implementable. Such γ (or rather the $\bar{r}$ associated with the control symbol $\tilde{r}$) represent feasible points of the inequality system. The identification of those $\bar{r}$ which yield a supervisable plant automaton can be viewed as an event identification problem. In this case, the event is the arc between two conjunctive events. The identification algorithm determines whether or not such an arc can be actually realized by the system.

The event identification problem is actually a problem in determining feasible points for inequality systems. There exist good numerical techniques for finding such points. One class of algorithms for determining such points uses the so-called method of centers to update a hypothesized feasible point. The method of centers computes a sequence of convex bodies and their centers in such a way that the computed centers converge to the feasible point. Depending on the analytic form of the convex bodies and the centers, different types of algorithms are obtained. Examples of such algorithms are the so-called ellipsoid method [17] and interior point methods based on logarithmic barrier functions [18].

The method of centers can be viewed as an inductive inference algorithm. Inductive inference is a machine learning algorithm in which a system learns a Boolean function from examples[19]. The learning algorithm is outlined below.

- Hypothesis: The hypothesis assumes that the conjunctive event $\mathbf{c}$ is supervisable by vector $\bar{r}_i = \gamma(\tilde{r}_i)$. This hypothesis is represented by a convex body K_i in which $\bar{r}_i$ is a "center" for K_i. Prior work [16] used ellipsoidal convex bodies and their associated geometric centers. It is quite possible and probably desirable to also use analtyic centers defined by logarithmic barrier functions. This latter approach has proven useful in interior point linear programming algorithms [18].
- Experiment: The inequalities in the proposition indicate that specific information has to

P. J. Antsaklis, M. D. Lemmon and J. A. Stiver, "Hybrid System Modeling and Event Identification,"
P roc. o f t he T hirty-First A nnual A llerton C o nference o n C ommunication C ontrol a nd C omputing ,
pp 64-73, Univ. of Illinois at Urbana-Champaign, Sept. 29-Oct. 1, 1993.

be gathered by the system before the sufficient conditions can be tested. The experimental component of the algorithm involves the measurement of these quantities. In particular, the system must be able to estimate or measure the Lie derivatives of the plant. As noted above, these derivatives can be estimated if there is a prior plant model. When no such model is available, there must be sensor capable of measuring or estimating these derivatives.

- Query: The third component of the algorithm is the oracle query. The oracle is a Boolean functional which outputs 0 if the experimental data is consistent with the inequality system and which outputs 1 if the data does not satisfy the inequality system.

- Update: If the oracle's response is 0, then nothing is done. If the oracle declares 1, then the current data is inconsistent with the hypothesis that $\bar{r}_i$ satisfies the inequality system. This means that the hypothesis has to be changed. An update algorithm is called to compute a new convex body K_i and its associated center. The center of the updated convex body is then taken to be the new control vector $\bar{r}_i$. Prior work examined the use of central cut ellipsoid algorithms [16] in performing this update step. More recent experimental work has implied that the use of analytic centers (rather than geometric centers) may provide a more efficient way of updating the hypothesis.

The advantage of using the method of centers in solving such problems is that these methods have provable polynomial complexity. There are well-known results on the convergence of such methods. For example, an idealized method of centers algorithm which uses the geometric center can be shown to converge in finite time [15]. It is further shown that this finite convergence time scales in a way which is independent of problem complexity. The problem with the idealized algorithm is that the computation of geometric centers for arbitrary convex bodies is impractical. The central-cut ellipsoid algorithm represents a more practical way of computing geometric centers. For this algorithm, it has been shown that the finite convergence time scales on the order of $o(n \ln L)$ where n is a measure of plant complexity (generally the number of control policies available to control the plant) and L is a measure of the volume of the feasible set that's being searched for [20] [16].

The significance of the preceding discussion is that it suggests that the controllability of individual events in the plant automaton can be decided quickly and efficiently using the method of centers. The scaling results suggest that the $o(n \ln L)$ convergence rate for the ellipsoid method can be improved upon considerably and that from a theoretical perspective this rate will be bounded below in a manner which is independent of plant complexity. This immediately suggests that more efficient approaches than the ellipsoid method can be used for finding feasible points. This fact is already well-known and interior point algorithms using analytic centers based on so-called logarithmic barrier functions [18] represent an important class of update procedures which may provide even more efficient identification of supervisable automaton events.

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

References

[1] J. Albus, et al, "Theory and practice of intelligent control", In *Proc. 23rd IEEE COMPCON*, pp. 19–39, 1981.

[2] P. J. Antsaklis, K. M. Passino, and S. J. Wang, "Towards intelligent autonomous control systems: Architecture and fundamental issues", *Journal of Intelligent and Robotic Systems*, vol. 1, pp. 315–342, 1989.

[3] P. J. Antsaklis, K. M. Passino, and S. J. Wang, "An introduction to autonomous control systems", *IEEE Control Systems Magazine*, vol. 11, no. 4, pp. 5–13, June 1991.

[4] P. J. Antsaklis and K. M. Passino, "Introduction to intelligent control systems with high degree of autonomy", In *An Introduction to Intelligent and Autonomous Control*, edited by P. J. Antsaklis and K. M. Passino, chapter 1, pp. 1–26. Kluwer Academic Publishers, 1993.

[5] P. J. Antsaklis and K. M. Passino, editors, *An Introduction to Intelligent and Autonomous Control*, Kluwer Academic Publishers, 1993, 448 p.

[6] "Special issue on autonomous intelligent machines", *IEEE Computer*, vol. 22, no. 6, June 1989.

[7] K. M. Passino and P. J. Antsaklis, "Modeling and analysis of artificially intelligent planning systems", In *Introduction to Intelligent and Autonomous Control*, edited by P.J.Antsaklis and K.M.Passino, chapter 8, pp. 191–214. Kluwer, 1993.

[8] G. N. Saridis, "Toward the realization of intelligent controls", *Proceedings of the IEEE*, vol. 67, no. 8, pp. 1115–1133, Aug. 1979.

[9] G. N. Saridis, "Foundations of the theory of intelligent controls", In *Proceedings of the IEEE Workshop on Intelligent Control*, pp. 23–28, 1985.

[10] G. N. Saridis, "Knowledge implementation: Structures of intelligent control systems", In *Proceedings of the IEEE International Symposium on Intelligent Control*, pp. 9–17, 1987.

[11] G. N. Saridis, "Analytic formulation of the principle of increasing precision with decreasing intelligence for intelligent machines", *Automatica*, vol. 25, no. 3, pp. 461–467, 1989.

[12] B. P. Zeigler, *Multifacetted Modelling and Discrete Event Simulation*, Acedemic Press, NY, 1984.

[13] J. A. Stiver and P. J. Antsaklis, "DES supervisor design for hybrid control systems", In *Proceedings of the Twenty-Ninth Annual Allerton Conference on Communication, Control, and Computing*, University of Illinois at Urbana-Champaign, Sep. 1993, To appear.

[14] 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.

[15] A.S. Nemirovsky and D.B. Yudin, *Problem Complexity and Method Efficiency in Optimization*, Wiley and Sons, 1983.

[16] 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.

[17] R.G. Bland, D. Goldfarb, and M.J. Todd, "The ellipsoid method: a survey", *Operations Research*, vol. 29, pp. 1039–1091, 1981.

[18] C.C. Gonzaga, "Path-following methods for linear programming", *SIAM Review*, vol. 34, no. 2, pp. 167–224, June 1992.

[19] D. Angluin and C.H. Smith, "Inductive inference: Theory and methods", *Computing Surveys*, vol. 15, no. 3, pp. 237–269, September 1983.

[20] Groetshel, Lovasz, and Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Berlin, 1988.