M.D. Lemmon and P.J. Antsaklis, "Inductively Inferring Valid Logical Models of Continuous-State Dynamical Systems," Technical Report of the ISIS Group, No. ISIS-93-010, Univ. of Notre Dame, December 1993.
Journal of Theoretical Computer Science, Amir Pnueli and Joseph Sifakis Eds., vol 138, Part 1, pp 201-210, February 1995

**Reprinted from**

# Theoretical Computer Science

Theoretical Computer Science 138 (1995) 201–210

# Inductively inferring valid logical models of continuous-state dynamical systems

Michael Lemmon*, Panos J. Antsaklis
*Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA*

ELSEVIER

ELSEVIER

Theoretical Computer Science 138 (1995) 201–210

Theoretical
Computer Science

# Inductively inferring valid logical models of continuous-state dynamical systems

Michael Lemmon*, Panos J. Antsaklis

*Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA*

**Abstract**

Hybrid control systems consist of a discrete event (DES) controller supervising a continuous-state (CSS) plant. A controller can be synthesized by obtaining a DES controller for an equivalent DES representation (DES plant) of the CSS plant. An important issue concerns the logical invariance (stability) of DES plant transitions to variations in the initial CSS plant state. This paper provides a set of sufficient conditions for the existence of stable transitions in the DES plant. For CSS plants which are affine in their control policies, these conditions form a system of linear inequalities over the space of control vectors used by the CSS plant. Feasible points to this inequality system are inductively determined using a method of centers algorithm known as the ellipsoid method.

## 1. Introduction

Hybrid control systems arise when a discrete event system (DES) is used to supervise the behavior of a continuous-state system (CSS). Such systems arise in the supervisory control of large-scale systems found in semiconductor manufacturing, power distribution, flexible manufacturing, and chemical process control. One approach [1, 7] to the design of hybrid control systems extracts a logical model of the continuous-state system's behavior. This logical model is called the *DES plant*. An important issue in this approach concerns the "stability" of the logical model represented by the DES plant. Ideally, the transitions in the DES plant should be unchanged by infinitesimal perturbations in the CSS plant's state. If, for example, the plant supervisor is obtained using the Ramadge–Wonham framework [6], it is extremely important that the DES plant be controllable. A system whose DES plant has "unstable" event transitions will generally not be controllable. A technique is therefore needed to identify "stable" transitions in DES plants.

This paper derives a set of sufficient conditions insuring the stability of transitions in the DES plant with respect to variations in the initial CSS plant state. These

* Corresponding author.


SSDI 0304-3975(94)00148-0

conditions are based on the Lyapunov stability theory and hence the notion of logically stable transitions is clearly related to conventional notions of continuous-state system stability. In particular, it is shown that for the class of systems studied by this paper (Section 2), that the stability of DES plant transitions will be guaranteed provided the control vectors driving the CSS plant satisfy a set of sufficient conditions (Section 3). When the CSS plant is affine in its control policies these conditions become a linear inequality system whose feasible points are the control vectors insuring stable logical transitions in the DES plant. These feasible points can then be found using conventional method of center algorithms (Section 4) such as the ellipsoid method. Such method of center algorithms can be realized as inductive learning procedures, thereby allowing the hybrid system to "learn" those control vectors insuring stable transitions in the DES plant.

## 2. Hybrid control systems

A hybrid control system consists of four interconnected subsystems; the plant, the controller, the actuator, and the generator. Fig. 1 illustrates the assumed interconnections. The plant is a continuous-state continuous-time dynamical system. The controller is a discrete event system (DES). Because both plant and controller evolve over distinctly different types of sets, an interface is required to connect the systems. This interface is assumed to consist of two subsystems, the actuator and generator. The actuator is responsible for transforming sequences of symbols issued by the DES controller into a continuous-time control signal. Similarly, the generator is responsible for transforming a continuous-time state trajectory into a sequence of plant symbols. The hybrid control system examined by this paper was originally discussed

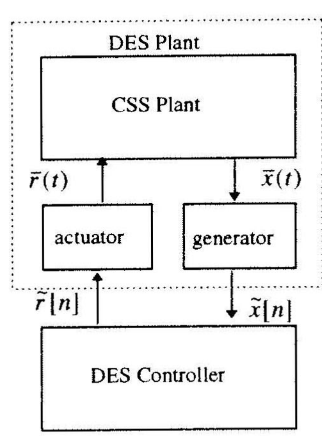


Fig. 1. Hybrid control system.

in [1]. In this modeling framework the interface generates a sequence of logical symbols describing the plant's symoblic behavior with regard to an underlying finite topology of the plant's state space.

The *continuous-state system plant* or *CSS plant* is represented by the ordered $p+1$-tuple,

$$\mathscr{P}_c = (\bar{X}, \Phi_t^{\bar{r}_1}, \ldots, \Phi_t^{\bar{r}_p}), \tag{1}$$

where $\bar{X} \subset \Re^n$ is the plant's state space and where $\Phi_t^{\bar{r}_i}: \bar{X} \to \bar{X}$ (for $i = 1, \ldots, p$) is the $i$th family (indexed by time, $t$) of transition operators generated by the differential equation

$$\dot{\bar{x}}(t) = f(\bar{x}(t), \bar{r}_i). \tag{2}$$

It is assumed that $\bar{r}_i$ is a constant vector in $\Re^m$ ($i = 1, \ldots, p$) and that $f: \Re^n \times \Re^m \to \Re^n$ is a Lipschitz continuous function serving as the infinitesimal generator of the transition operators $\Phi_t^{\bar{r}_1}$.

An example of such a CSS plant is given by the following set of $p$ differential equations

$$\dot{\bar{x}}^{(i)} = f_0(\bar{x}^{(i)}) + \sum_{j=1}^{m} r_{ji} f_j(\bar{x}^{(i)}), \tag{3}$$

where $i = 1, \ldots, p$, $\bar{x}^{(i)}$ is the state vector for the $i$th differential equation, and $r_{ji}$ ($j = 1, \ldots, m$) are the components of the $i$th equation's control vector, $r_i$. In this case, the collection of mappings $f_j: \Re^n \to \Re^n$ for $j = 0, \ldots, m$ represents a set of $m+1$ *control policies*. The control policies of the $i$th differential equation are linearly mixed by the components of the control vector, $\bar{r}_i$. Such a nonlinear system is often referred to as being "affine" in its control vectors. Under suitable assumptions it includes the class of nonlinear systems which can be linearized through appropriate feedback.

The controller is a discrete event system called the *supervisor*. It receives as inputs a sequence, $\tilde{x}[n]$, of symbols drawn from a finite alphabet $\tilde{X}$ of *plant symbols*. The supervisor outputs a sequence, $\tilde{r}[n]$, of symbols drawn from a finite alphabet $\tilde{R}$ of *control symbols*. The controller's dynamics are assumed to be modeled by a deterministic finite automaton (DFA).

The interface *generator* is that part of the interface which transforms the CSS plant's state trajectory, $\bar{x}(t)$, into a sequence of plant symbols, $\tilde{x}[n]$. Let $\bar{X} \subset \Re^n$ be the CSS plant's state space and let $\tilde{X}$ be a finite alphabet of $q$ plant state symbols. Let $\mathscr{B}$ denote a finite collection of $q$ disjoint sets which partition the CSS plant's state space. Denote the $i$th element of $\mathscr{B}$ as $\boldsymbol{b}_i$, $i = 1, \ldots, q$. $\mathscr{B}$ will be called the hybrid system's *generator basis*. Elements of $\mathscr{B}$ will be called *generator sets*. The generator's output is assumed to take the following form:

$$\tilde{x}[n] = \alpha(\bar{x}(\tau_x[n])), \tag{4}$$

where $\alpha : \bar{X} \to \tilde{X}$ is a mapping taking a state vector in the CSS plant's state space onto the plant symbol alphabet. In particular, this *generator mapping* is defined such that $\alpha(\bar{x}) = \tilde{x}_i$ which $\tilde{x}_i \in \tilde{X}$ if and only if $\bar{x} \in \boldsymbol{b}_i$. $\tau_x[n]$, is a sequence of times (measured with respect to the CSS plant's clock) when the state space trajectory crosses into the open set $\boldsymbol{b}_i$. In practice, it would be assumed that the elements of $\mathscr{B}$ are given beforehand. For example, if it is known that certain subsets of the CSS plant's state space are forbidden, then these sets should be contained in $\mathscr{B}$. Another example is found in model predictive control. In this case, the generator sets might contain setpoints about which the CSS plant must be regulated. Assuming that $\mathscr{B}$ is given, then the CSS plant's desired behavior would be a formal specification on how the CSS plant should transition between elements of $\mathscr{B}$. The problem considered in this paper concerns the determination of an interface actuator which insures that such transitions occur in a controllable (or rather stable) manner.

The interface *actuator* is that part of the interface which transforms a sequence of DES supervisory control symbols, $\tilde{r}[n]$, into a trajectory of control vectors $\bar{r}(t)$ used to drive the plant. It is assumed that the output of the actuator is given by the following equation:

$$\bar{r}(t) = \sum_{n=0}^{\infty} \gamma(\tilde{r}[n]) I(t, \tau_r[n], \tau_r[n+1]), \tag{5}$$

where $\gamma : \tilde{R} \to \bar{R}$ is a mapping which takes the control symbol at discrete time $n$, $\tilde{r}[n]$, onto a constant control vector in $\bar{R}$. Associated with this sequence of control symbols, is a sequence of times (measured with respect to the CSS plant's clock), $\tau_r[n]$, when the $n$th control symbol was issued by the supervisor. For causality reasons, it will be assumed that these times are related to the generator times $\tau_x[n]$ as $\tau_x[n] \leqslant \tau_r[n] \leqslant \tau_x[n+1]$. The indicator function $I(t, \tau_r[n], \tau_r[n+1])$ is unity for all $t$ in the interval $(\tau_r[n], \tau_r[n+1])$ and zero elsewhere. The control trajectory, $\bar{r}(t)$, is therefore a piecewise constant function of time.

Note from Fig. 1 that the combination of plant and interface behave like a discrete event system. The combined plant/interface system is referred to as the *DES plant*. The inputs to the DES plant are the sequence of *control events*, $(\tilde{r}[n], \tau_r[n])$. The outputs of the DES plant are the sequence of *plant events*, $(\tilde{x}[n], \tau_x[n])$. Note that the "events" are ordered pairs consisting of a symbol and a time associated with that symbol. The *DES plant* of a hybrid control system is the labeled digraph $\mathscr{P}_d = (\tilde{X}, A)$. The set of vertices, $\tilde{X}$, is the alphabet of plant symbols. The arcs, $A$, form a subset of $\tilde{X} \times \tilde{X}$. Each arc is labeled by a control event, $(\tilde{r}, \tau_r) \in \tilde{R} \times \Re$, and a plant event, $(\tilde{x}, \tau_x) \in \tilde{X} \times \Re$.

## 3. Transition stability

The DES plant can be seen as a "logical" interpretation of the CSS plant's symbolic behavior. If that interpretation accurately and reliably predicts the CSS plant's behavior, then it is a valid interpretation or *model* of the CSS plant. As one approach [7] to hybrid control system design involves synthesizing DES controllers for an extracted DES plant, the "validity" of the DES plant is an issue which must be addressed.

DES plant validity can be viewed in terms of the invariance of plant and control event sequences to small perturbations in the CSS plant's state. An arc of the DES plant represents a transition of the CSS plant's state between two subsets of the CSS plant's state space. The labeling of that arc represents the symbolic behavior of that transition. A valid DES plant would preserve that labeling under small perturbations of the initial CSS plant state. This viewpoint is formalized in the following definition of hybrid system transition or $T$-stability.

**Definition 1.** Let $\mathscr{P}_d=(\tilde{X}, A)$ be a DES plant for a hybrid control system. Let $\tilde{x}_i$ and $\tilde{x}_j$ be two vertices in $\tilde{X}$ with associated generator sets $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$, respectively. Consider the arc $(\tilde{x}_i, \tilde{x}_j)\in A$ labeled with control event, $(\tilde{r}, \tau_r)$ and plant event, $(\tilde{x}, \tau_x)$. Let $\bar{r}=\gamma(\tilde{r})$ be the CSS plant control vector associated with control symbol $\tilde{r}$ through the interface actuator mapping $\gamma$.

The arc $(\tilde{x}_i, \tilde{x}_j)$ is transition or $T$-stable if only if for all $\bar{x}_0\in\boldsymbol{b}_i$, there exists an open neighborhood, $N_\varepsilon(\bar{x}_0)$, centered at $\bar{x}_0$ and a finite time $0<T<\infty$ such that the set

$$N_T=\{\bar{x}_T\colon \bar{x}_T=\Phi_T^{\bar{r}}(\bar{x}),\ \bar{x}\in N_\varepsilon(\bar{x}_0)\} \tag{6}$$

is an open subset of $\boldsymbol{b}_j$ and such that the plant symbol, $\tilde{x}$, issued during the transition is identical for all transitions starting in $N_\varepsilon(\bar{x}_0)$ and ending in $N_T$.

If the CSS plant is affine in its control policies then the following proposition provides a set of sufficient conditions for a single arc of the DES plant to be $T$-stable. The following proposition makes use of the directional or Lie derivative of a functional. Let $V\colon\Re^n\to\Re$ be a continuously differentiable functional and let $f\colon\Re^n\to\Re^n$ be a smooth vector field over $\Re^n$. Let $[\nabla_{\bar{x}} V]$ denote the gradient vector of $V$. The Lie (directional) derivative of $V$, $L_fV\colon\Re^n\to\Re$, with respect to vector field $f$ is given by $L_fV=[\nabla_{\bar{x}} V]^t f$. This is the inner product of the gradient of $V$ with the vector field $f$. With the preceding definitions the following proposition can now be stated.

**Proposition 1.** *Consider a DES plant, $\mathscr{P}_d=(\tilde{X}, A)$ of a hybrid system whose CSS plant is affine in its control policies (Eq. (3)). Consider an arc $(\tilde{x}_j, \tilde{x}_k)$ of the DES plant with control event label, $(\tilde{r}, \tau)$. Let $\bar{r}=(r_1, \ldots, r_m)^t=\gamma(\tilde{r})$. This arc will be T-stable if there exists a set of continuously differentiable positive definite functionals $V_i\colon\Re^n\to\Re$ $(i=1, \ldots, q)$ which are zero on a closed proper subset of $\boldsymbol{b}_i\in\mathscr{B}(i=1, \ldots, q)$ such that for all $\bar{x}\in\bar{X}$*

*and* $\bar{x} \notin \boldsymbol{b}_k$,

$$\begin{pmatrix} L_{f_0}V_k & L_{f_1}V_k & \cdots & L_{f_m}V_k \end{pmatrix} \begin{pmatrix} 1 \\ r_1 \\ \vdots \\ r_m \end{pmatrix} < 0 \tag{7}$$

*and for all* $\bar{x} \in \boldsymbol{b}_i$ *for* $i = 1, \ldots, q$ *and* $i \neq k$,

$$\begin{pmatrix} L_{f_0}V_i & L_{f_1}V_i & \cdots & L_{f_m}V_i \end{pmatrix} \begin{pmatrix} 1 \\ r_1 \\ \vdots \\ r_m \end{pmatrix} > 0. \tag{8}$$

**Proof.** Note that a sufficient condition for the transition to be $T$-stable is that all trajectories starting in $\boldsymbol{b}_j$ are attracted to $\boldsymbol{b}_k$ and are repelled by any other elements of $\mathscr{B}$. This condition is satisfied provided $\boldsymbol{b}_k$ contains a global attractor for the controlled system and all other $\boldsymbol{b}_i$ ($i \neq k$) are repellors. These conditions can easily be established by constructing a Lyapunov functional [4] over the state space such that the system is globally stable to $\boldsymbol{b}_k$. The LaSalle invariance principle [4] can be used for this purpose and immediately yields the first conditions given in the proposition. To insure that all other sets are repelling, it is sufficient to guarantee that the functional $V_i$ for these generator sets always force the state trajectory out of the set. The second condition of the proposition guarantees this behavior. □

How easily such conditions can be satisfied will depend on the generator sets in $\mathscr{B}$. It can be shown, for example, that if the sets in $\mathscr{B}$ form a finite partition of the state space, then the conditions in Proposition 1 will generally be impossible to satisfy unless the boundaries of all generator sets lie on integral manifolds of the distribution of control policies. If the generator basis $\mathscr{B}$, does not form a complete partition of the state space, then the conditions are much easier to satisfy.

The preceding proof relies on the generator sets forming global attractors and repellors for the CSS plant. This condition is clearly not necessary for $T$-stability. For many situations, this condition may only have to hold in a local sense. Even so, however, the proposition is very valuable. Dynamical systems are always subjected to unpredicted external disturbances which may force the plant state off the controlled trajectory. When such disturbances occur, it is desirable that the transition remain "stable". One way to insure this is to require that the generator sets be global attractors and repellors. Therefore, while the condition in Proposition 1 is restrictive, it provides a test which is useful in the face of unmodeled CSS plant disturbances. In addition to this (as will be seen below) the proposition provides a practical condition for testing the $T$-stability of a DES plant.

## 4. Inductively inferring $T$-stable interfaces

The sufficient conditions obtained in Proposition 1 pertain to a single transition arc, $(\tilde{x}_i, \tilde{x}_j)$ of the DES plant. These conditions form a system of linear inequality constraints in the CSS plant's control space, $\bar{R}$. Feasible points satisfying the inequality system are therefore constant control vectors $\bar{r} \in \bar{R}$ which guarantee that the single arc is $T$-stable. By finding the feasible points for each arc in the DES plant, a set of control vectors $\bar{r}_i$ (where $i = 1, \ldots, p$) associated with the control symbols $\tilde{r}_i$ is obtained. The systematic application of this approach to every arc in a given DES plant can then be used to determine an actuator mapping, $\gamma$, which $T$-stabilizes the entire DES plant.

Deciding the $T$-stability of the entire DES plant can only be done if there exists a numerically efficient method for finding feasible points. One class of algorithms for doing this is the class of *method of centers* [5] algorithms. Method of center algorithms compute a sequence of convex bodies and their centers in such a way that the computed centers converge to a feasible point. Depending on the analytic form of the convex bodies and the centers, different types of algorithms are obtained. A particularly well-known example is the so-called ellipsoid method [2]. In this algorithm, the convex bodies are ellipsoidal sets containing the set of feasible vectors and the centers are the geometric centers of these ellipsoids. In the following examples, we will illustrate the use of the ellipsoid method as an inductive learning algorithm.

The ellipsoid method (as all method of center algorithms) is an iterative algorithm. The initialization of the algorithm determines a convex set (ellipsoid) which is known to contain the feasible set. The center of this bounding convex body is then computed and is tested for feasibility using the given inequality constraints. If the center is not feasible, then half of the ellipsoidal set can be discarded to form a smaller convex body containing the set of feasible gains. The ellipsoid method computes the minimum volume ellipsoid containing this smaller convex body. The updated ellipsoid has a smaller volume than the preceding bounding ellipsoid so that the algorithm iteratively computes a tighter and tighter upper bound on the set of feasible points. It is well known that the ellipsoid method will converge after a finite number of "failed" tests. This finite number of failures is bounded above in a way which grows on the order of $O(m \ln L)$ where $L$ is the volume of the feasible set and $m$ is the problem's dimensionality (in our case the number of available control policies). Consequently, the ellipsoid algorithm converges in a finite time that scales in a polynomial manner with problem complexity and the size of the feasible set.

Ellipsoid and method of center algorithms are usually implemented as off-line procedures in which the entire inequality system is always available for testing a given center's feasibility. Recall that the inequalities in Proposition 1 check the sign of the inner product of the control vector $\bar{r}$ with a vector of Lie derivatives. The inequality must hold for all CSS plant states and therefore represents an infinite number of inequality constraints. In many cases it is possible to measure or estimate these directional (Lie) derivative vectors as the CSS plant state evolves over time. This

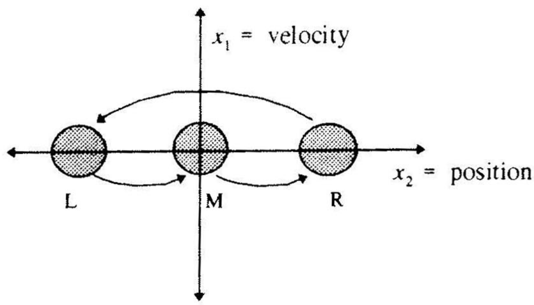


Fig. 2. Generator sets (L, M, and R) for the double integrator example. The specified DES plant transitions are shown by the arrows.

suggests that rather than implementing the ellipsoid method as an off-line procedure, it can be implemented as an on-line procedure thereby yielding an adaptive or inductive learning procedure for deciding the $T$-stability of the DES plant. Assume that the CSS plant is attempting to transition the hybrid system between two distinct generator sets. We can measure the CSS plant state and its Lie derivatives as the hybrid system attempts to realize this DES plant arc. The measured data is then used to evaluate the feasibility of the control vector $\bar{r}$. If the control vector is declared infeasible, then a new control vector is computed. Otherwise, the control vector is not modified. After this test, the system is allowed to evolve further and another CSS plant state (Lie derivative) is measured. The feasibility of the control is checked again and updated if needed. This process of measuring, testing, and updating is then repeated until a feasible control vector is found. Note that all of the convergence results for the off-line ellipsoid method are applicable to the on-line method. It can therefore be concluded that this on-line or inductive learning algorithm must converge after a *finite number* of *failed tests* provided such the feasible set is not empty [3]. Since there is a known upper bound on this convergence time, we can conclude that the $T$-stability of a given arc is decidable after a finite number of updates.

The following example illustrates how the learning algorithm would be used to decide whether or not a given DES plant specification can be realized. In this example, we are given a continuous-state plant (double integrator) with a finite number of linearly mixed control policies. The system equations are

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}0 & 0\\ 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+r_1\begin{bmatrix}x_1\\ 0\end{bmatrix}+r_2\begin{bmatrix}x_2\\ 0\end{bmatrix}+r_3\begin{bmatrix}1\\ 0\end{bmatrix}, \tag{9}$$

where the state vector is $\bar{x}=(x_1,x_2)^t$. It is assumed that the generator basis $\mathscr{B}$ consists of sets $\boldsymbol{b}_i=\{\bar{x}\colon \|\bar{x}-\bar{x}_i\|^2<1\}$ for $i=1,2,3$ and where $\bar{x}_1=(0,0)^t$, $\bar{x}_2=(10,0)^t$, and $\bar{x}_3=(-10,0)^t$. DES plant symbols are therefore generated when the CSS plant state enters one of three disjoint hyperspheres in the state space (see Fig. 2).

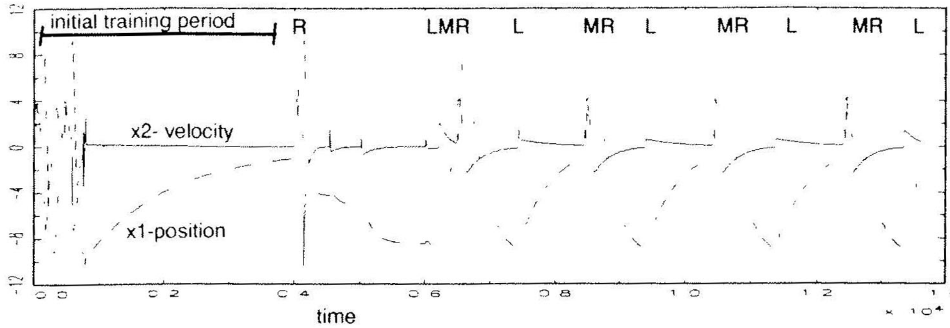


Fig. 3. CSS plant state trajectories for the double integrator example.

The objective is to use the inductive learning algorithm to determine the actuator mapping $\gamma$ which $T$-stabilizes a specified DES plant. The DES plant specification in this example is that the CSS plant state have the following transition. $\boldsymbol{b}_1 \to \boldsymbol{b}_2$, $\boldsymbol{b}_2 \to \boldsymbol{b}_3$, and $\boldsymbol{b}_3 \to \boldsymbol{b}_1$. All other transitions within the DES plant are forbidden. The specified DES plant is shown in Fig. 2.

To implement the inductive learning algorithm, the CSS plant is first started in $\boldsymbol{b}_1$ and the control vector $\bar{r}$ associated with the $\boldsymbol{b}_1 \to \boldsymbol{b}_2$ transition is applied. At regular time intervals, the plant states are measured, the associated Lie derivatives of the system are computed, and the inequality constraints tested to see whether or not $\bar{r}$ is feasible. In this simulation example, the inequality system was generated with respect to a quadratic Lyapunov functional, $V_i = \|\bar{x} - \bar{x}_i\|^2$ $(i = 1, 2, 3)$. If the test decleares $\bar{r}$ to be infeasible, then $\bar{r}$ is updated using the central cut ellipsoid method. The testing of $\bar{r}$ continues in this way until the system enters another generator set. Upon entering another generator set ($\boldsymbol{b}_2$ for instance), the control vector associated with the specified transition from this set is issued and then inductively tested (and updated). This process then continues until none of the control vectors associated with the specified DES plant transitions are updated anymore.

Fig. 3 shows the system's state history as the on-line inference procedure is used. The figure shows the two states of the system evolving over time. Also marked on the plot are the sequence of plant symbols (L, M, R) issued by the hybrid system's generator. The figure shows that after an initial transient period, in which the on-line algorithm is searching for the $T$-stabilizing control vectors of the specified DES plant, the state trajectories settle down to a stable limit cycle whose logical behavior is that of the specified DES plant.

## 5. Summary

This paper has presented a method for inductively learning interfaces of hybrid control systems which insure that a given DES plant has "stable" transitions. In this paper, transition stability was interpreted as the invariance of the transition labelings with respect to perturbations of the initial CSS plant state. This notion of transition or $T$-stability was shown to be closely tied to conventional notions of continuous-state system Lyapunov stability and it was this relationship which provided the basis for inductively deciding and determining $T$-stabilizing control vectors for the hybrid system.

## Acknowledgment

The partial financial support of the National Science Foundation (IRI91-09298 and MSS92-16559) is gratefully acknowledged.

## References

[1] P.J. Antsaklis, J.A. Stiver and M.D. Lemmon, Hybrid system modeling and autonomous control systems, in: R.L. Grossman, A. Nerode, A. Ravn and H. Rischel, eds., *Hybrid Systems*, Lecture Notes in Computer Science, Vol. 736, (Springer, Berlin 1993) 366–392.
[2] R.G. Bland, D. Goldfarb and M.J. Todd, The ellipsoid method: a survey, *Oper. Res.* **29** (1981) 1039–1091.
[3] M. Groetshel, L. Lovasz and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization* (Springer, Berlin, 1988).
[4] J.P. LaSalle and S. Lefschetz, *Stability by Lyapunov's Direct Method* (Academic Press, New York, 1961).
[5] A.S. Nemirovsky and D.B. Yudin, *Problem Complexity and Method Efficiency in Optimization* (Wiley, New York, 1983).
[6] P. Ramadge and W.M. Wonham, The control of discrete event systems, *Proc. IEEE* **77** (1) (1989) 81–89.
[7] J.A. Stiver and P.J. Antsaklis, DES supervisor design for hybrid control systems. *Proc. 31st Ann. Allerton Conf. on Communication Control and Computing*. Univ. of Illinois at Urbana-Champaign (1993).