Supervisory Control Using Computationally Efficient Linear Techniques: A Tutorial Introduction

Technical Report of the ISIS Group
at the University of Notre Dame
ISIS-97-015
October, 1997

John O. Moody and Panos J. Antsaklis
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556

Interdisciplinary Studies of Intelligent Systems

Supervisory Control Using Computationally Efficient Linear Techniques: A Tutorial Introduction[1 2]

Abstract

This paper provides an overview of a computationally efficient method for synthesizing supervisory controllers for discrete event systems (DES). The DES plant and controller are described by Petri nets which provide a useful linear algebraic model for both control analysis and synthesis. It is shown how a set of linear constraints on the plant's behavior can be enforced, accounting for possibly uncontrollable or unobservable transitions in the plant net, using techniques from Petri net theory, integer programming, and linear systems. The paper is written as a tutorial introduction to the approach. Several results presented here have been reported elsewhere in the literature.

1 Introduction

A methodology to automatically derive feedback supervisory controllers for discrete event systems (DES) described by Petri nets appears in [13]. The control designer is presented with a Petri net model of a DES and a set of linear constraints on the state space of the DES. The control goal is to insure that the constraints are met during the plant's normal operation. In the spirit of supervisory control, this task is accomplished by prohibiting certain occurrences in the plant which would cause one or more of the constraints to be violated. The method is based on the idea that specifications representing desired plant behaviors can be enforced by making them invariants of the controlled Petri net. The resulting controllers are themselves Petri nets and are identical to the monitors [2] of Giua et al. The controller's size is proportional to the number of constraints.

The supervisor is used to enforce a set of linear constraints on the state space of the plant DES. These constraints are not as general as the languages enforced by Ramadge and Wonham [10] in their work on supervisory control using automata, but the solution algorithms are simpler, and they can be used to describe a broad variety of problems including

- A large range of forbidden state problems.
- Serial, parallel and general mutual exclusion problems.
- A class of logical predicates on plant behavior [12].

[1]This technical report has also appeared as J. O. Moody and P. J. Antsaklis, "Supervisory control using computationally efficient linear techniques: A tutorial introduction", In *Proceedings of the 5th IEEE Mediterranean Conference on Control and Systems*, Session MP1, Paphos, Cyprus, July 1997.

[2]This research was partially funded by the National Science Foundation. Grant ECS95–31485.

- Conditions involving the occurrence of events and particular regions of the state space.
- Conditions involving the concurrence of events.
- The modeling of shared resources [6].

The approach was extended in [7] to apply to Petri nets which contain uncontrollable transitions, the firing of which cannot be inhibited by the controller. This work was partially motivated by the research of Li and Wonham [3] dealing with the enforcement of linear constraints on vector discrete event systems with uncontrollable events. The approach in [7] was expanded in [4] to include uncontrollable and unobservable transitions in a unified framework. Algorithms were presented for automatically computing new sets of plant constraints which accounted for uncontrollable and unobservable transitions while still enforcing the original constraints. Unobservable transitions force a special structure on the Petri net controller which can be used to characterize valid controllers and simplify controller design. These results appear in [5]. These contributions extend the applicability of the control method while maintaining its original emphasis: they also relate to Petri net place invariants and are again simple to implement with excellent numerical properties.

The paper is structured as follows. The controller synthesis method for plants with controllable transitions is described in section 2. A methods for dealing with uncontrollable and unobservable transitions is covered in section 3. An example is used to illustrate the method in section 4, and concluding remarks are given in section 5.

2 Automatic Controller Synthesis

The system to be controlled is modeled by a Petri net with n places and m transitions and is known as the process or *plant net*. The incidence matrix of the plant net is D_p. It is assumed that all the enabled transitions can fire. It is possible that the process net will violate certain constraints placed on its behavior, thus the need for control. The *controller net* is a Petri net with incidence matrix D_c made up of the process net's transitions and a separate set of places. The *controlled system* or *controlled net* is the Petri net with incidence matrix D made up of both the original process net and the added controller. The control goal is to force the process to obey constraints of the form

$$L\mu_p \leq b \tag{1}$$

where μ_p is the marking vector of the Petri net modeling the process, L is an $n_c \times n$ integer matrix, b is an n_c dimensional integer vector and n_c is the number of constraints. Note that the inequality is with respect to the individual elements of the two vectors $L\mu_p$ and b and can be thought of as the logical conjunction of the individual "less than or equal to" constraints. This definition will be used throughout this paper whenever vectors appear on either side of an inequality sign.

Inequality (1) can be transformed into an equality by introducing an external Petri net controller which contains places which represent nonnegative "slack variables" μ_c. The constraint then becomes

$$L\mu_p + \mu_c = b \tag{2}$$

where μ_c is an n_c dimensional integer vector which represents the marking of the controller places. Note that $\mu_c \geq 0$ because the number of tokens in a place can not become negative; thus equation (2) implies inequality (1). The controller places insure that the weighted sums of tokens in the process net's places are always less than or equal to the elements of b. The places which maintain the inequality constraints are part of a separate net called the controller net. The structure of the controller net will be computed by observing that the introduction of the slack variables forces a set of place invariants on the overall controlled system defined by equation (2).

Place invariants are one of the structural properties of Petri nets. See [8, 9, 11] for more information on Petri nets and their properties and analysis. A place invariant is defined as every integer vector x which satisfies

$$x^T\mu = x^T\mu_0 \text{ (a constant)} \tag{3}$$

where μ_0 is the net's initial marking, and μ represents any subsequent marking. Equation (3) means that the weighted sum of the tokens in the places of the invariant remains constant at all markings and this sum is determined by the initial marking of the Petri net. The place invariants of a net are elements of the kernel of the net's incidence matrix, i.e., they can be computed by finding integer solutions to

$$x^T D = 0 \tag{4}$$

where D is an $n \times m$ incidence matrix with n being the number of places and m the number of transitions.

The matrix D_c contains the arcs that connect the controller places to the transitions of the process net. Let $\mathbb{Z}$ be the set of integers. The incidence matrix $D \in \mathbb{Z}^{(n+n_c)\times m}$ of the closed loop system is given by

$$D = \begin{bmatrix} D_p \\ D_c \end{bmatrix} \tag{5}$$

and the marking vector $\mu \in \mathbb{Z}^{n+n_c}$ and initial marking μ_0 are given by

$$\mu = \begin{bmatrix} \mu_p \\ \mu_c \end{bmatrix} \qquad \mu_0 = \begin{bmatrix} \mu_{p_0} \\ \mu_{c_0} \end{bmatrix} \tag{6}$$

Note that equation (2) is in the form of (3), thus the invariants defined by equation (2) on the system (5), (6) must satisfy equation (4).

$$\begin{aligned} X^T D = [L \;\; I] \begin{bmatrix} D_p \\ D_c \end{bmatrix} &= 0 \\ LD_p + D_c &= 0 \end{aligned} \tag{7}$$

where I is an $n_c \times n_c$ identity matrix since the coefficients of the slack variables in equation (2) are all equal to 1. The following proposition follows from this discussion.

Proposition 1. The Petri net controller, $D_c \in \mathbb{Z}^{n_c \times m}$ with initial marking μ_{c_0}, which enforces constraints (1) when included in the closed loop system (5) with marking (6) is defined by

$$D_c = -LD_p \tag{8}$$

with initial marking

$$\mu_{c_0} = b - L\mu_{p_0} \tag{9}$$

assuming that the transitions with arcs from D_c are controllable, observable, and that $\mu_{c0} \geq 0$.

The controller defined by proposition 1 is maximally permissive, assuming that all transitions are controllable and observable, in that it will never disable a transition that would not directly violate the constraints if fired. The proof of this result is given in [13].

Proposition 1 creates a controller which will enable and inhibit various transitions in the plant. If any of these transitions are uncontrollable or unobservable, then the controller defined by this method may be invalid. The next section shows how a transformation of the constraints can be performed in order to avoid these transitions while still enforcing the original constraints.

3 Handling Uncontrollable and Unobservable Transitions

Consider the situation where the controller is not allowed to influence certain transitions in the plant Petri net. These transitions are called uncontrollable. It is illegal for the Petri net controller to include an arc from one of the controller places to any of these uncontrollable plant transitions, since these kinds of connections can lead to the disabling of plant transitions.

Equation (8) in section 2 shows that it is possible to construct the incidence matrix D_c of a maximally permissive Petri net controller as a linear combination of the rows of the incidence matrix of the plant. Negative elements in D_c correspond to arcs from controller places to plant transitions. These arcs act to inhibit plant transitions when the corresponding controller places are empty, and thus they can only be applied to plant transitions which permit such external control. Group all of the columns of D_p which correspond to transitions which can not be controlled into the matrix D_{uc}. The matrix LD_{uc} must contain no positive elements[3], as these will correspond to controlling arcs when constructing the supervisor

[3] Actually LD_{uc} may contain positive elements when the controller is merely observing uncontrollable transitions and not inhibiting them, but this situation is not covered here.

$D_c = -LD_p$. An enforceable set of constraints will satisfy

$$LD_{\mathrm{uc}} \leq 0 \tag{10}$$

It is also possible that transitions within the plant may be unobservable, i.e., they are defined on the Petri net graph because they represent the occurrence of real events, but these events are either impossible or too expensive to detect directly. It is also possible, in the event of a sensor failure, that a transition might suddenly become unobservable, forcing a redesign or adaptation of the control law. It is illegal for the controller to change its state based on the firing of an unobservable transition, because there is no direct way for the controller to be told that such a transition has fired. Both input and output arcs from the controller places are used to change the controller state based on the firings of plant transitions. Let the matrix D_{uo} represent the incidence matrix of the unobservable portion of the Petri net. This matrix is composed of the columns of D_p which correspond to unobservable transitions, just as D_{uc} is composed of the uncontrollable columns of D_p. It is illegal for the controller $D_c = -LD_p$ to contain any arcs in the unobservable portion of the net, thus an enforceable set of constraints will satisfy

$$LD_{\mathrm{uo}} = 0 \tag{11}$$

Conditions (10) and (11) indicate that it is possible to observe a transition that we can not inhibit, but it is illegal to directly inhibit a transition that we can not observe.

Suppose, given a set of constraints $L\mu_p \leq b$, we construct the matrices LD_{uc} and LD_{uo} and observe that there are violations to conditions (10) and/or (11). Since the controller is made of a linear combination of the rows of D_p, it is interesting to consider the situation where we use the addition of further rows from D_{uc} in order to eliminate the positive elements of LD_{uc} and use rows from D_{uo} to eliminate the nonzero elements of LD_{uo}, i.e., if we are going to use a place invariant forming Petri net controller, what additions to the constraints would we need to make in order to eliminate positive elements from LD_{uc} and nonzero elements from LD_{uo}? What constraints, of the form $L'\mu_p \leq b'$, that can be enforced by an invariant-based controller, will also maintain the original constraint $L\mu_p \leq b$ while not interfering with the uncontrollable/unobservable portions of the plant? The following lemma appeared in [7].

Lemma 2.

$$\text{Let } R_1 \in \mathbb{Z}^{n_c \times m} \text{ satisfy } R_1\mu_p \geq 0 \ \forall \ \mu_p. \tag{12}$$

$$\begin{array}{c}\text{Let } R_2 \in \mathbb{Z}^{n_c \times n_c} \\ \text{positive definite diagonal matrix}\end{array} \tag{13}$$

If $L'\mu_p \leq b'$ where

$$L' = R_1 + R_2 L \tag{14}$$

$$b' = R_2(b + \mathbf{1}) - \mathbf{1} \tag{15}$$

and $\mathbf{1}$ is an n_c dimensional vector of 1's, then $L\mu_p \leq b$.

Lemma 2 shows a class of constraints, $L'\mu_p \leq b'$, which, if enforced, will imply that $L\mu_p \leq b$ are also enforced. In [4], Lemma 2 is used to prove a portion of the following proposition.

Proposition 3. Let a plant Petri net with incidence matrix D_p be given with a set of uncontrollable transitions described by D_{uc} and a set of unobservable transitions described by D_{uo}. A set of linear constraints on the net marking, $L\mu_p \leq b$, are to be imposed. Assume R_1 and R_2 meet (12) and (13) with $R_1 + R_2 L \neq 0$ and let

$$\begin{array}{c} \begin{bmatrix} R_1 & R_2 \end{bmatrix} \begin{bmatrix} D_{\mathrm{uc}} & D_{\mathrm{uo}} & -D_{\mathrm{uo}} & \mu_{p_0} \\ LD_{\mathrm{uc}} & LD_{\mathrm{uo}} & -LD_{\mathrm{uo}} & L\mu_{p_0} - b - \mathbf{1} \end{bmatrix} \\ \leq \begin{bmatrix} 0 & 0 & 0 & -\mathbf{1} \end{bmatrix} \end{array} \tag{16}$$

Then the controller

$$D_c = -(R_1 + R_2 L) D_p = -L' D_p \tag{17}$$

$$\mu_{c_0} = R_2(b + \mathbf{1}) - \mathbf{1} - (R_1 + R_2 L)\mu_{p_0} = b' - L'\mu_{p_0} \tag{18}$$

exists and causes all subsequent markings of the closed loop system (5),(6) to satisfy the constraint $L\mu_p \leq b$ without attempting to inhibit uncontrollable transitions and without detecting unobservable transitions.

The usefulness of proposition 3 for specifying controllers to handle plants with uncontrollable and unobservable transitions lies in the ease in which the matrices R_1 and R_2, with the appropriate properties, can be generated. Algorithms for solving this problem including a method involving matrix row operations and by solving a linear integer programming problem appear in [4]. The method of using row operations is outlined below, but instead of presenting the pseudo code algorithms of [4], the overall motivation and goals of the method are described.

To meet assumption (12), it is sufficient to assume that all of the elements of R_1 are nonnegative, since the elements of μ_p are nonnegative by definition. In general, for unbounded μ_p, it is necessary that all of the elements of R_1 be nonnegative, however if bounds on μ_p are known, then it is possible to generate valid R_1 vectors which contain some negative elements. If R_1 and R_2 which satisfy (12) and (13) do exist, then they can be found by performing row operations on $\begin{bmatrix} D_{\mathrm{uc}} \\ LD_{\mathrm{uc}} \end{bmatrix}$ and $\begin{bmatrix} D_{\mathrm{uo}} \\ LD_{\mathrm{uo}} \end{bmatrix}$. Row operations act as premultiplications of a matrix, just as $\begin{bmatrix} R_1 & R_2 \end{bmatrix}$ premultiplies these two matrices in inequality (16). R_1 and R_2 can be found by finding a set of row operations which do not involve premultiplication of any row by a negative number and which force the LD_{uc} portion of the matrix to contain all zero or negative elements and the LD_{uo} matrix to be all zeros. Note that assumption (13), which requires R_2 to be a positive definite matrix, is not restrictive. This matrix simply represents the premultiplication coefficients of the rows of the LD_{uc} and LD_{uo} portions of the matrices undergoing row operations. We can assume this matrix is diagonal because LD_{uc} and LD_{uo} are linearly dependent with D_{uc} and D_{uo}, i.e., we will never need to take linear combinations of the rows in LD_{uc} or LD_{uo}. We can also assume that the diagonal elements are positive since, if negative numbers are required, they can be accounted for by R_1, which still needs to meet assumption (12). This technique is illustrated for a plant with uncontrollable transitions in the following section.

4 Example – The Unreliable Machine

We now provide a simple example in order to illustrate the concepts that have been covered above. The example plant is partially based on the model of an "unreliable machine" from [1]. The machine is used to process parts from an input queue, completed parts are moved to an output queue. The machine is considered unreliable because it is possible that it may break down and damage a part during operation. This behavior is captured in the plant model. Damaged parts are moved to a separate queue from the queue for successfully completed parts. The Petri net model of the plant is shown in figure 1, and a description of the various places and transitions is given in table 1.

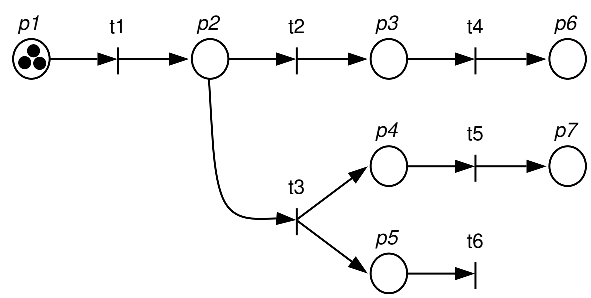

Figure 1: Petri net model of an uncontrolled unreliable machine.

The plant model has two uncontrollable transitions, t_2 and t_3. Transition t_3 represents machine break down and so obviously can not be controlled. Transition t_2 is considered uncontrollable because the controller can not force the machine to instantly finish a part that is not yet completed, nor does it direct the machine to stop working on an unfinished part. The transition is labeled uncontrollable in order to prevent a control design from attempting either of these two actions.

Places

p_1	Input queue – Number of parts remaining
p_2	Machine is busy, part is being processed
p_3	Waiting for transfer to completed parts queue
p_4	Waiting for transfer to damaged parts queue
p_5	Machine is waiting to be repaired
p_6	Completed parts queue
p_7	Damaged parts queue

Transitions

t_1	Part moves from input queue to machine
t_2	*Uncontrollable:* Part processing is complete
t_3	*Uncontrollable:* Machine fails, part is damaged
t_4	Part moves to completed parts queue
t_5	Part moves to damaged parts queue
t_6	Machine is repaired

Table 1: Place and transition descriptions for the Petri net of figure 1.

4.1 Controller Synthesis

The Petri net model of the plant has the following incidence matrix and marking vector.

$$D_p = \left[\begin{array}{c} \begin{array}{cccccc} -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \underbrace{0 \quad\; 0}_{D_{\mathrm{uc}}} & & 0 & 1 & 0 \end{array} \end{array} \right] \quad \mu_p = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \end{bmatrix} \tag{19}$$

The initial conditions are $\mu_{p_0} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$.

If the machine is broken, we do not want to load a new part until repairs have been completed. This means that places p_2 and p_5 should contain at most one token:

$$\mu_2 + \mu_5 \leq 1 \tag{20}$$

Parts waiting to be transferred to a storage queue, whether completed or damaged, wait in the same position on the machine. The Petri net model uses two places, p_3 and p_4, to represent waiting parts, because there are two different destinations. In order to prevent conflict, the second constraint is

$$\mu_3 + \mu_4 \leq 1 \tag{21}$$

Using the matrix form of constraint (1) we have

$$\underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}}_{L} \mu_p \leq \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{b} \tag{22}$$

First we must check the uncontrollability condition.

$$LD_{\mathrm{uc}} = \begin{bmatrix} -1 & 0 \\ 1 & 1 \end{bmatrix}$$

We need all of the elements of LD_{uc} to be less than or equal to zero if we are to avoid using uncontrollable transitions. There is no problem with the first row, but a transformation will have to be found to eliminate the 1's in the second row. This can be done by applying row operations from the matrix D_{uc} to eliminate the positive elements in the second row of LD_{uc}.

$$\begin{array}{c} \begin{bmatrix} 0 & 0 \\ -1 & -1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \\ \begin{bmatrix} 1 & 1 \end{bmatrix} \end{array} \quad \begin{array}{c} \text{Row } 8 = \text{Row } 8 + \text{Row } 2 \\ \Longrightarrow \end{array} \quad \begin{array}{c} \begin{bmatrix} 0 & 0 \\ -1 & -1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & 0 \end{bmatrix} \end{array}$$

Because constraint (20) required no transformation, the first row of R_1 will be all zeros. A row operation involving the addition of the second row of the D_{uc} matrix is required to transform constraint (21), thus the second row of R_1 will be all zeros with a one in the second column. It was not necessary to premultiply either constraint, thus R_2 will be an identity matrix.

$$R_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad R_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

We now apply equations (14) and (15) to find the transformed constraints represented by L' and b'.

$$\underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}}_{L'} \mu_p \leq \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{b'}$$

The controller is the calculated using equations (17) and (18).

$$D_c = -L'D_p = \left[\begin{array}{cccccc} -1 & 1 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 1 & 1 & 0 \end{array} \right]$$

$$\mu_{c_0} = b' - L'\mu_{p_0} = \left[\begin{array}{c} 1 \\ 1 \end{array} \right]$$

The controlled net is shown in figure 2. The constraint logic is enforced and no input arcs are drawn to the uncontrollable transitions.

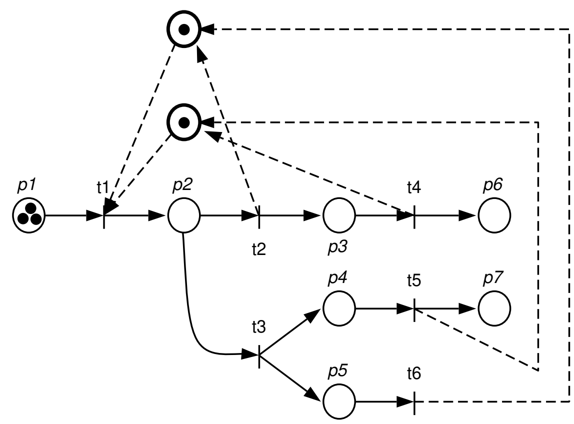

Figure 2: The controlled unreliable machine.

4.2 Discussion

An extensive look at many of the issues central to this research can be found in the work of Li and Wonham [3]. These authors show that optimal, or maximally permissive, control actions which account for uncontrollable transitions can be found by repeated applications of a linear integer programming problem (LIP), assuming that valid control actions actually exist and that the uncontrollable portion of the net contains no loops. They also give sufficient conditions under which the solution to the LIP has a closed form expression. These conditions place a certain tree structure on the uncontrollable portion of the net. When this tree structure is further limited, Li and Wonham are able to prove that the optimal control law which insures $L\mu_p \leq b$ can be written $C\mu_p \leq d$. This is the case where it is possible to represent the action of the optimal control law with ordinary Petri nets. In this situation, it is possible to find R_1 and R_2 by performing row operations on $\left[\begin{array}{c} D_{\mathrm{uc}} \\ LD_{\mathrm{uc}} \end{array} \right]$ which is much more desirable, computationally, than analytically solving an LIP. However the tree structure assumed by Li and Wonham is only sufficient, not necessary, for example, the structure of the uncontrollable part of the plant in section 4 does not conform to Li and Wonham's "type 2 tree structure," however an optimal solution was found and implemented using an ordinary Petri net controller. There are also cases where, following the procedures presented above, suboptimal Petri net controllers may be derived. These suboptimal controllers may be sufficient for many tasks, depending on the application.

5 Conclusions

This paper has presented computationally efficient methods for constructing feedback controllers for ordinary Petri nets, even in the face of uncontrollable and unobservable plant transitions. The method is based on the idea that specifications representing desired plant behaviors can be enforced by making them invariants of the controlled net, and that simple row operations on a matrix containing the uncontrollable and unobservable columns of the plant incidence matrix can be used to eliminate controller use of illegal transitions.

The significance of this particular approach to Petri net controller design is that the control net can be computed very efficiently, thus the method shows promise for controlling large, complex systems, or for recomputing the control law online due to some plant failure.

There are several areas of ongoing research for this work. Necessary and sufficient conditions for a linear control law to be maximally permissive in the face of uncontrollable and unobservable transitions are not known. Time is becoming an increasingly important factor in the area of DES control. Ordinary Petri nets are sufficient for modeling sequences in time and concurrency, but it may be desirable to extend the method for use with actual timed Petri nets. It may also be possible to extend the applicability of the method by expanding the kinds of constraints that may be enforced. Methods for transforming logical predicates on the plant behavior into linear inequality constraints, and a class of nonlinear constraints, are currently being explored.

References

[1] A. A. Desrochers and R. Y. Al-Jaar, *Applications of Petri Nets in Manufacturing Systems*, IEEE Press, Piscataway, NJ, 1995.

[2] A. Giua, F. DiCesare, and M. Silva, "Generalized mutual exclusion constraints on nets with uncontrollable transitions", In *Proceedings of the 1992 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 974–979, Chicago, IL, October 1992.

[3] Y. Li and W. M. Wonham, "Control of vector discrete event systems II – controller synthesis", *IEEE Transactions on Automatic Control*, vol. 39, no. 3, pp. 512–530, March 1994.

[4] J. O. Moody and P. J. Antsaklis, "Supervisory control of Petri nets with uncontrollable/unobservable transitions", In *Proceedings of the 35th IEEE Conference on Decision and Control*, pp. 4433–4438, Kobe, Japan, December 1996.

[5] J. O. Moody and P. J. Antsaklis, "Characterization of feasible controls for Petri nets with unobservable transitions", In *Proceedings of the 1997 American Control Conference*, volume 4, pp. 2354–2358, Albuquerque, New Mexico, June 1997.

[6] J. O. Moody, P. J. Antsaklis, and M. D. Lemmon, "Automated design of a Petri net feedback controller for a robotic assembly cell", In *Proceedings of 1995 INRIA/IEEE Symposium on Emerging Technologies and Factory Automation*, volume 2, pp. 117–128, Paris, France, October 1995.

[7] J. O. Moody, P. J. Antsaklis, and M. D. Lemmon, "Feedback Petri net control design in the presence of uncontrollable transitions", In *Proceedings of the 34th IEEE Conference on Decision and Control*, volume 1, pp. 905–906, New Orleans, LA, December 1995.

[8] T. Murata, "Petri nets: Properties, analysis, and applications", *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[9] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice Hall, Engelwood Cliffs, NJ, 1981.

[10] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems", *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–97, 1989.

[11] W. Reisig, *Petri Nets*, Springer-Verlag, Berlin; New York, 1985.

[12] K. Yamalidou and J. C. Kantor, "Modeling and optimal control of discrete-event chemical processes using Petri nets", *Computers in Chemical Engineering*, vol. 15, no. 7, pp. 503–519, 1991.

[13] K. Yamalidou, J. O. Moody, M. D. Lemmon, and P. J. Antsaklis, "Feedback control of Petri nets based on place invariants", *Automatica*, vol. 32, no. 1, pp. 15–28, January 1996.