AUTOMATED DESIGN OF A PETRI NET FEEDBACK CONTROLLER FOR A ROBOTIC ASSEMBLY CELL

John O. Moody, Panos J. Antsaklis, and Michael D. Lemmon

Department of Electrical Engineering
University of Notre Dame, Notre Dame, IN 46556
Email: jmoody@maddog.ee.nd.edu

1 Introduction

In this paper a method for computing a feedback controller for a manufacturing assembly cell modeled by a Petri net will be presented and a method for dealing with uncontrollable and unobservable transitions is proposed. The control goal is to enforce a set of linear constraints on the marking behavior or state of the Petri net. The method discussed here is a powerful means of realizing these constraints because it is simple to calculate, and the Petri net structure of the solution makes the controller easy to implement. The controller computation is derived using the concept of Petri net place invariants. This derivation originally appeared in [5] and produces controllers identical to the monitors [2] of Giua et al.

This approach has been recently extended to apply to Petri nets which contain uncontrollable transitions, the firing of which cannot be forced to or prevented from occurring if they are currently enabled. This extension also relates to place invariants and is easy and transparent to implement with excellent numerical properties. This invariant based approach to design of Petri net feedback controllers has a number of advantages:

1. The design method is transparent as it is based on the concept of place invariants.
2. The resulting controller and consequently the overall controlled system are described by standard Petri nets where a variety of tools for analysis, both graphical and algebraic, are available. Verification of the design is therefore rather straightforward.
3. The design method has excellent numerical properties. This make this control design approach particularly appealing in control reconfiguration applications where because of a failure the controller must be redesigned on line.

Of course the method also has limitations and they are discussed in [2, 5].

The paper is organized as follows. Sections 2 and 3 outline automatic methods to generate Petri net plant controllers and to modify base constraints to account for uncontrollable plant transitions. A piston rod assembly robotic manufacturing cell plant model is presented in section 4. The control techniques presented in sections

2 and 3 are used to control the plant model. Section 5 contains a discussion of how the plant controller can deal with the management of finite resources. The idea of uncontrollable transitions is extended to unobservable transitions in section 6 where a new controller is designed to account for a sensor failure in the assembly cell. Section 7 contains some brief concluding remarks.

2 Automatic Controller Synthesis from Plant Constraints

The system to be controlled is modeled by a Petri net with n places and m transitions and is known as the plant or *process net.* The incidence matrix of the process net is D_p. It is assumed that all 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 goal is to force the process to obey n_c linear 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.

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

$$D = \begin{bmatrix} D_p \\ D_c \end{bmatrix} \tag{2}$$

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

$$\mu = \begin{bmatrix} \mu_p \\ \mu_c \end{bmatrix} \qquad \mu_0 = \begin{bmatrix} \mu_{p_0} \\ \mu_{c_0} \end{bmatrix} \tag{3}$$

A maximally permissive Petri net controller [2, 5] which enforces constraints (1) when included in the closed loop system (2) with marking (3) is defined by the incidence matrix

$$D_c = -LD_p \tag{4}$$

and the initial marking

$$\mu_{c_0} = b - L\mu_{p_0} \tag{5}$$

assuming that the transitions with input arcs from D_c are controllable and that the constraints are not contradictory, i.e. $\mu_{c_0} \geq 0$. The derivation of this controller is given in [5] and is based on forcing place invariants on the closed loop system.

This controller will enable and inhibit various transitions in the plant. If any of these transitions are uncontrollable then the controller defined by this method is invalid. The next section discusses how a transformation of the constraints can be performed so that the uncontrollable transitions in the net receive no input arcs from the controller places.

3 Handling Uncontrollable 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.

Let D_{uc} be the incidence matrix of the uncontrollable portion of the process net. D_{uc} is composed of the columns of D_p which correspond to the uncontrollable transitions. Recall that, assuming no self loops, positive elements in an incidence matrix refer to arcs from transitions to places, and negative elements refer to arcs from places to transitions. $D_{uc} \in Z^{m \times n_u}$ where n_u is the number of uncontrollable transitions. Given a set of constraints, $L\mu \leq b$, the Petri net controller given by $D_c = -LD_p$ violates the uncontrollability constraint if LD_{uc} contains any elements greater than zero. The uncontrollability constraint dictates that we can not draw any arcs from the controller places to the transitions, and the portion of the controller corresponding to the uncontrollable transitions is given by $-LD_{uc}$, thus the elements of LD_{uc} must all be less than or equal to zero.

Suppose we are unable to meet the uncontrollability constraint, i.e. we have positive values in the matrix LD_{uc}. It is necessary to transform the constraint vector L such that the original constraint of $L\mu_p \leq b$ is still maintained, while obeying the uncontrollability constraint. The constraint transformation will take the form

$$L'\mu_p \leq b' \tag{6}$$

where

$$\begin{aligned} L' &= R_1 + R_2 L \qquad (7)\\ b' &= R_2(b+\mathbf{1}) - \mathbf{1} \qquad (8)\end{aligned}$$

$R_1 \in Z^{n_c \times m}$, $R_2 \in Z^{n_c \times n_c}$, $\mathbf{1}$ is an n_c dimensional vector of 1's and

$$R_1\mu_p \geq 0 \qquad \text{for all possible } \mu_p \tag{9}$$

$$R_2 \text{ is a positive definite diagonal matrix} \tag{10}$$

It can be shown that any controller which enforces $L'\mu_p \leq b'$ will also enforce $L\mu_p \leq b$.

In order to meet the uncontrollability constraint we need $L'D_{uc} \leq 0$ which will insure that the controller contains no arcs leading to the uncontrollable transitions in the plant. We need

$$\begin{aligned} R_1 D_{uc} + R_2 L D_{uc} &\leq 0 \qquad \text{or} \\ \begin{bmatrix} R_1 & R_2 \end{bmatrix} \begin{bmatrix} D_{uc} \\ LD_{uc} \end{bmatrix} &\leq 0 \qquad (11)\end{aligned}$$

Thus if R_1 and R_2 can be found which satisfy inequality (11) and satisfy assumptions (9) and (10) then a controller can be synthesized which enforces $L\mu_p \leq b$ and does not contain any input arcs to the uncontrollable transitions in the plant. In general R_1 and R_2 are found by performing row operations on $\begin{bmatrix} D_{uc} \\ LD_{uc} \end{bmatrix}$. This procedure is demonstrated in the following sections.

4 Piston Rod Robotic Assembly Cell

The automatic control concepts discussed above are illustrated here using the example of a piston rod assembly cell which is taken from chapter 8 of [1]. The Petri net model of the plant is shown in Fig. 1. Table 1 details the meaning of each place in the net. A token in any of the Petri net places signifies that the action or condition specified in Table 1 is taking place. The piston rod assembly is performed by two robots, and the primary feedback mechanism is a vision system. An S-380 robot is used to prepare and allign the parts for assembly, and an M-1 robot installs the cap on the piston rod. The specific duties of each robot are described below.

S-380: The S-380 robot remains idle until a new engine block and crank shaft become available. This event is represented by the appearance of a token in place p_1 in Fig. 1. The firing of transition t_1 indicates the start of the process. At this time the S-380 moves the crank shaft into allignment and brings a new piston rod into the work area. These actions are represented by places p_2 and p_3. The firing of transition t_3 indicates that the S-380 has completed its duties for the particular engine block.

M-1: The M-1 robot starts its duties by picking up a piston pulling tool (place p_4) and, assuming the S-380 has brought a piston rod into position. pulls the piston rod into the engine block and replaces the pulling tool (place p_5). The M-1 then picks up a cap and secures it to the piston rod using two bolts (places p_6 and p_7). The firing of transition t_8 indicates that the M-1 has successfully installed the cap and the engine block has been conveyed out of the work space. At this time work can begin on a new engine block.

p_1	Work area clear, engine block and crank shaft ready to be processed.
p_2	S-380 robot alligns the crank shaft.
p_3	S-380 robot picks up new piston rod and positions it in work space.
p_4	M-1 robot picks up the piston pulling tool.
p_5	M-1 robot pulls the piston rod into the engine block, returns pulling tool.
p_6	M-1 robot picks up a cap and positions it on piston rod.
p_7	M-1 robot readies two bolts and uses them to fasten cap to piston rod.

Table 1: Place descriptions for the piston rod assembly Petri net of Fig. 1.

The incidence matrix, D_p, and initial marking, μ_{p0}, of the plant are given by

$$D_p = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \qquad \mu_{p0} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{12}$$

There are a number of constraints that must be imposed on this assembly cell in order to properly synchronize the robots and to insure that physical limitations are obeyed. There is only one S-380 and one M-1 robot available for the assembly process, thus only one of each robot can be working at any given time. Places p_2

J.O. Moody, P.J. Antsaklis and M.D. Lemmon, "Automated Design of a Petri Net Feedback Controller for a Robotic Assembly Cell," I NRIA/IEEE C o nference o n E merging T echnologies a nd F actory A utomatio n , Paris, France, October 10-13, 1995.

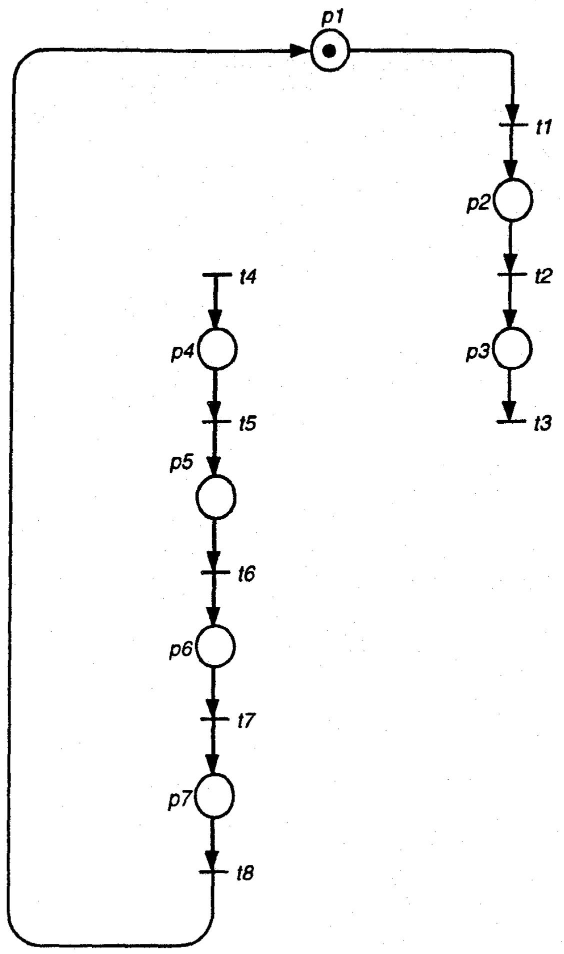

Figure 1: The base Petri net model for the piston rod robotic assembly cell.

and p_3 represent activity for the S-380 and places p_4, p_5, p_6, and p_7 represent activity for the M-1, thus

$$\mu_2 + \mu_3 \leq 1 \tag{13}$$

$$\mu_4 + \mu_5 + \mu_6 + \mu_7 \leq 1 \tag{14}$$

The plant graph already insures that the S-380 will not begin its task until an engine block and crank shaft are available and the work space is clear, however we need to make sure that the M-1 does not try to pull the piston rod into the engine until the S-380 has finished alligning the crank shaft and readying a new piston rod. In other words, we must insure that transition t_5 does not fire until after transition t_3 has fired. A review of the graph structure of the plant net shows that we can write this constraint as

$$\mu_1 + \mu_2 + \mu_3 + \mu_5 + \mu_6 + \mu_7 \leq 1 \tag{15}$$

Note that this constraint allows the M-1 robot to ready its pulling tool, the task of p_4, at the same time that the S-380 robot is performing one of its tasks.

There are several finite resources that are used in the assembly process. A piston rod is readied at p_3, a pulling tool is required at p_4 and p_5, a cap is required at p_6.

and two nuts are used at p_7. The plant controller will need to know if any of these resources is unavailable in order to stall operation until the parts are ready. It is possible to provide the controller with the necessary hooks to manage these resources by associating a constraint with each of the places (or sets of places) that requires the use of a finite resource:

$$\mu_3 \leq 1 \tag{16}$$
$$\mu_4 + \mu_5 \leq 1 \tag{17}$$
$$\mu_6 \leq 1 \tag{18}$$
$$\mu_7 \leq 1 \tag{19}$$

The precise reason for these constraints and the way they may be used to manage the finite resources is elaborated on in section 5.

One final constraint placed on the assembly cell involves the smooth and uninterrupted operation of the M-1 robot. Perhaps due to the nature of the M-1 robot's dynamics or programming, we would prefer that its operation not be interrupted from the point that it pulls the piston rod into the engine block until it has completed fastening the cap on the piston rod. We can model this constraint by marking transitions t_6, t_7 and t_8 as uncontrollable. Thus we have told the controller that once a token passes into p_5, it can not stall the plant's progress until the token has passed on to places p_6, p_7 and completion. This means that if the controller does want to stall the M-1 robot, it should do it before it starts its primary operation.

Constraints (13) – (19) are now combined so that all of the constraints can be expressed in the single matrix inequality

$$\underbrace{\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{L} \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \end{bmatrix} \leq \underbrace{\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}}_{b} \tag{20}$$

The controller which enforces these constraints will have seven places (one for each row of L), but first we must insure that our constraints meet the uncontrollability condition. Transitions t_6, t_7, and t_8 are uncontrollable, so the matrix D_{uc} is composed of the last three columns of D_p. According to section 3, the matrix LD_{uc} must be checked for any positive values, which would indicate that the current constraints would violate the uncontrollability conditions.

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

Rows 6 and 7 of LD_{uc} contain positive values, thus it will be necessary to manipulate the sixth and seventh constraints (constraints (18) and (19)) so that their enforcement will not generate a controller with arcs directed toward the uncontrollable transitions. Based on the procedure outlined in section 3, the offending rows of LD_{uc} can be eliminated by adding rows from D_{uc}. Keeping track of the row operations performed will yield the matrices R_1 and R_2 which will be used to generate the transformed constraint matrix L'. The row operations are as follows.

$$\begin{array}{c}\text{Rows 6 and 7}\\ \text{of } LD_{uc}\end{array} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \begin{array}{l}\text{Add row 5 of } D_{uc} \\ \text{Add row 6 of } D_{uc}\end{array} \begin{array}{c}\Rightarrow\\ \Rightarrow\end{array} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix}$$

$$\text{Add row 5 of } D_{uc} \Rightarrow \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Now that the matrix no longer contains any positive numbers, we can use the row operations performed to create the matrices R_1 and R_2:

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

Constraints (18) and (19) have been transformed as follows:

$$\mu_6 \leq 1 \Rightarrow \mu_5 + \mu_6 \leq 1 \tag{21}$$

$$\mu_7 \leq 1 \Rightarrow \mu_5 + \mu_6 + \mu_7 \leq 1 \tag{22}$$

The new constraint matrix is given by

$$L' = R_1 + R_2 L = \begin{bmatrix} 0&1&1&0&0&0&0 \\ 0&0&0&1&1&1&1 \\ 1&1&1&0&1&1&1 \\ 0&0&1&0&0&0&0 \\ 0&0&0&1&1&0&0 \\ 0&0&0&0&1&1&0 \\ 0&0&0&0&1&1&1 \end{bmatrix}$$

and the incidence matrix and initial marking of the controller are calculated according to equations (4) and (5):

$$Dc = -L'D_p = \begin{bmatrix} -1&0&1&0&0&0&0&0 \\ 0&0&0&-1&0&0&0&1 \\ 0&0&1&0&-1&0&0&0 \\ 0&-1&1&0&0&0&0&0 \\ 0&0&0&-1&0&1&0&0 \\ 0&0&0&0&-1&0&1&0 \\ 0&0&0&0&-1&0&0&1 \end{bmatrix} \qquad \mu_{c0} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

The controlled net is shown in Fig. 2 with the controller arcs shown as dashed lines and the controller places highlighted in bold. Table 2 describes the meaning of each of the controller places when a token is present within them. Note that place c_3 insures that the M-1 robot will wait to start working on the engine block until the S-380 has completed its task, but it is capable of readying the piston pulling tool while the S-380 is working. Also note that the controller directs no arcs to the uncontrollable transitions, however it still manages to enforce constraints (18) and (19).

c_1	S-380 robot is available for work.
c_2	M-1 robot is available for work.
c_3	S-380 robot has completed preparations, M-1 may begin.
c_4	A piston rod is available.
c_5	The piston pulling tool is available.
c_6	A cap is available.
c_7	Two nuts are available.

Table 2: Descriptions of the controller places in the Petri net of Fig. 2.

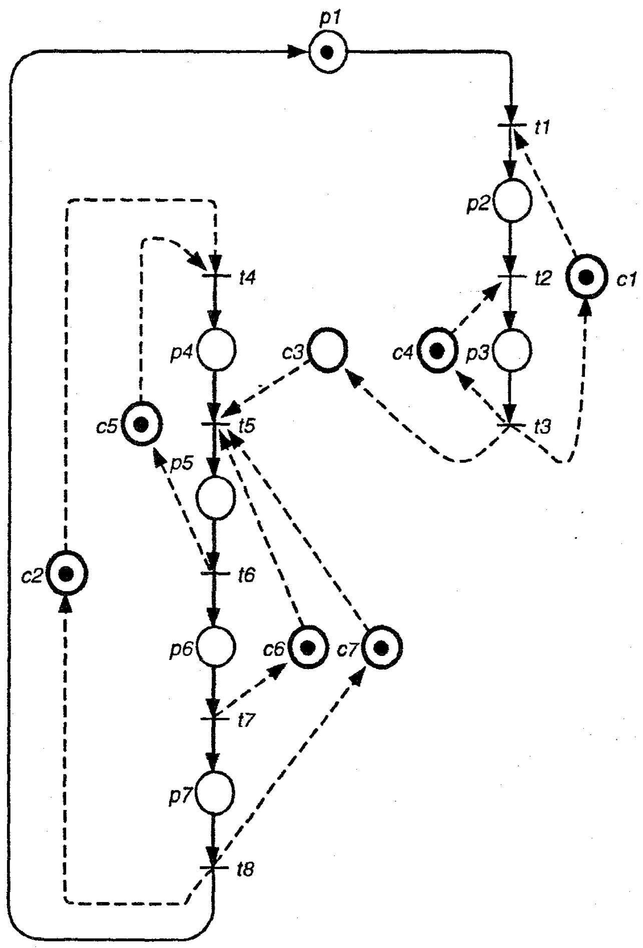

Figure 2: The assembly cell model with Petri net controller.

5 Using The Limited Resource Control Places

An examination of the enforced constraints represented by the matrix L' shows that some of these constraints are actually implied by others. For example, constraint (16) says that there can be at most one token in place p_3, but constraint (13) says that there can be at most one token, at any given time, in places p_2 and p_3. If constraint (13) is enforced then constraint (16) can not be violated, so why not remove constraint (16) as redundant? The answer is that constraint (16) and its associated controller place c_4 are not actually being used to prohibit a forbidden state, but to provide the controller with a means of dealing with finite resources, in this case, the availability of a new piston rod. This is the same situation with constraints (17), (18), and (19) and their associated places c_5, c_6, and c_7.

Suppose that one of the resources required by the assembly cell was no longer available. It would be possible for the computer running the assembly cell controller to eliminate the token in the corresponding place. For example, if there were no more piston rods available, then elimination of the token in place c_4 would prevent the S-380 robot from attempting to pick one up and bring it to the work space, instead it would wait for a piston rod to become available. When a piston rod did become available, a token could be placed in c_4 and the assembly could continue. Though this scheme would work, it is not very elegant from the point of view of Petri nets, i.e., tokens should not appear and disappear from a net without the corresponding firing of transitions. Fig. 3 shows how the resource controller places can be augmented with two uncontrollable transitions and a place in order to model the loss of finite resources while maintaining the standard Petri net framework of the model. Under normal operation the token in the resource place is used to permit the firing of a plant transition just as in Fig. 2. However the loss of the finite resource now corresponds to the firing of an uncontrollable transitions which robs the "resource is available" place of its token and stalls the operation of the plant. Another uncontrollable transition is then used to replace the missing token when the resource once again becomes available.

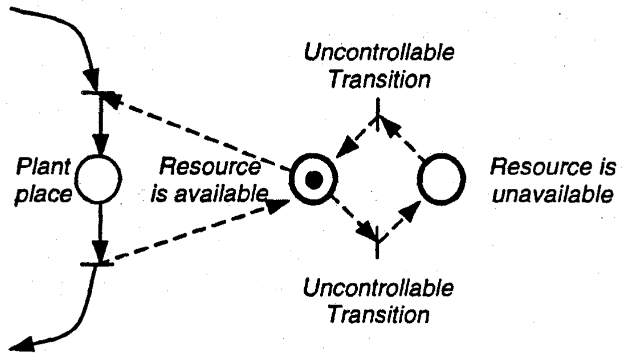

Figure 3: Modeling the loss of a finite resource using uncontrollable transitions.

6 Handling Sensor Failures

Uncontrollable transitions can be used to model the progress of irreversible processes, they can be used to prohibit undesirable control actions, as in section 4, they can represent the occurence of unpredictable events, as in section 5, or they can be used to model the failure of an actuator. In the case of an actuator failure a controllable transition is replaced with an uncontrollable one which requires some compensation on the part of the controller or the plant to be shut down. In this section, an analagous concept to uncontrollability, unobservability, will be explored. Unobservable transitions are analagous to uncontrollable transitions both in use and how they are handled analytically. Unobservable transitions may represent events which the controller can not detect, or which are too expensive or inconvenient to detect. An observable transition may be replaced with an unobservable transition in the case of a sensor failure, and this is the situation which will be used to illustrate these concepts here.

The piston rod assembly cell presented in [1] uses a vision system to provide sensory feedback to the controller. Suppose that an obstruction has appeared between the camera and the work space, partially obscuring the view of the M-1 robot's area. The controller can still observe the M-1 robot starting and completing its task, but it can no longer track the robot while it performs its duties. Transitions t_5, t_6, and t_7 have become unobservable. This means that there should be no arcs from any of these transitions to the controller places. Let D_{uo} be a matrix composed of the unobservable columns of D_p, in this example D_{uo} is composed of the fifth, sixth, and seventh columns of D_p. Unobservable transitions are handled analogously to uncontrollable ones by checking whether $L'D_{uo}$ contains any negative numbers. If $L'D_{uo}$ does contain any negative numbers then we will perform row operations on $L'D_{uo}$ to create a new matrix L'' which obeys the constraints. The new controller incidence matrix will the be given by $D_c = -L''D_p$.

First we observe that the controller needs to be modified to meet the unobservability constraint.

$$L'D_{uo} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}$$

The fifth and sixth rows of $L'D_{uo}$, corresponding to constraints (17) and (21), will need to be modified to eliminate the two -1's.

$$\begin{array}{c} \text{Rows 5 and 6} \\ \text{of } L'D_{uo} \end{array} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{array}{l} \text{Add rows 6 and 7 of } D_{uo} \\ \text{Add row 7 of } D_{uo} \end{array} \begin{array}{c} \Rightarrow \\ \Rightarrow \end{array} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

which creates the following transformations in the constraints:

$$\mu_4 + \mu_5 \leq 1 \;\Rightarrow\; \mu_4 + \mu_5 + \mu_6 + \mu_7 \leq 1 \qquad (23)$$
$$\mu + 5 + \mu_6 \leq 1 \;\Rightarrow\; \mu_5 + \mu_6 + \mu_7 \leq 1 \qquad (24)$$

The new constraint matrix corresponding to these row operations is

$$L'' = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

The Petri net corresponding to $Dc = -L''D_p$ is shown in Fig. 4. The unobservable transitions do not contain arcs to the controller places. Note that the condition of unobservability has caused places c_2 and c_5 to perform the same function. The same is true for places c_6 and c_7. These places should not be considered redundant because, even though they perform the same functions in the Petri net model, they represent different resources as discussed in section 5.

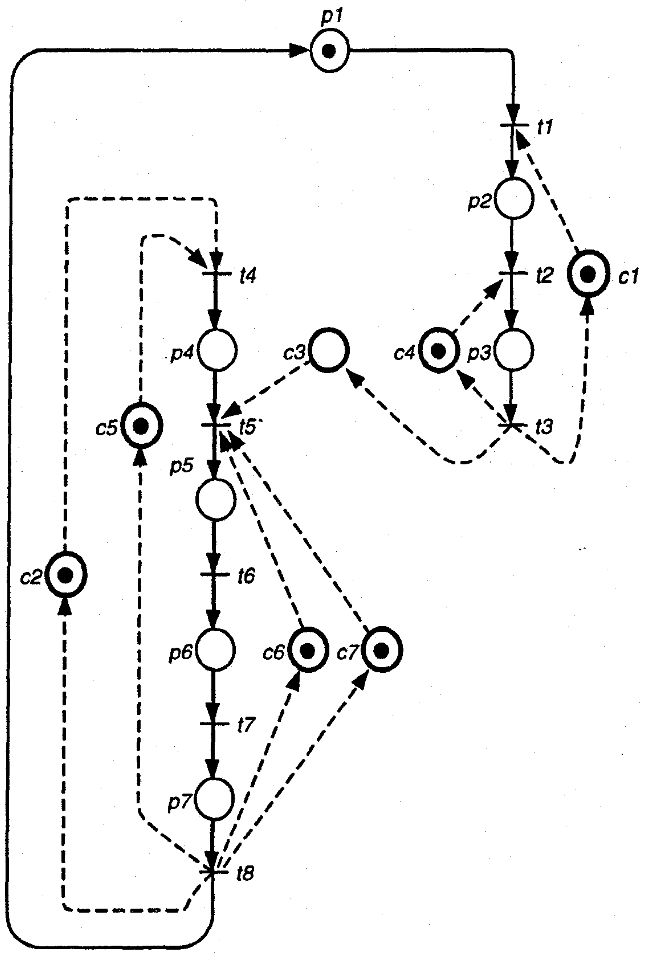

Figure 4: The assembly cell model with a controller that accounts for a sensor loss making transitions t_5, t_6 and t_7 unobservable.

7 Conclusions

This paper has presented an extended example demonstrating a particularly simple and computationally efficient method for constructing feedback controllers for untimed 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/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 efficiently and automatically based on the plant constraints. The method shows promise for controlling large, complex systems, or for recomputing control laws online due to some plant failure, such as the loss of a required resource, the break down of an actuator, or the corruption of a sensor.

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", *Proceedings of the 1992 IEEE International Conference on Systems, Man, and Cybernetics*, Chicago, IL, pp. 974–979, October 1992.

[3] Y. Li and W. M. Wonham, "Control of Vector Discrete Event Systems I – The Base Model", *IEEE Transactions on Automatic Control*, Vol 38, No. 8, August 1993.

[4] Y. Li and W. M. Wonham, "Control of Vector Discrete-Event Systems II – Controller Synthesis," *IEEE Transactions on Automatic Control*, Vol. 39, No. 3, March 1994.

[5] J. O. Moody, K. Yamalidou, M. Lemmon, and P. J. Antsaklis, "Feedback Control of Petri Nets Based on Place Invariants", *Proceedings of the 33rd Conference on Decision and Control*, Vol. 3, pp 3104–3109, Lake Buena Vista, FL, December 1994.

[6] T. Murata, "Petri Nets: Properties, Analysis, and Applications," *Proceedings of the IEEE*, vol 77, No. 4, pp. 541–580, 1989.

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

[8] 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, January 1989.

[9] W. Reisig, *Petri Nets*, Springer-Verlag, 1985.

[10] W. M. Wonham and P. J. G. Ramadge, "On the Supremal Controllable Sublanguage of a Given Language", *SIAM J. Control Optim.*, vol. 25, no. 3 pp 637-659, 1987.