Feedback Control of Petri Nets Based on Place Invariants

Technical Report of the ISIS Group
at the University of Notre Dame
ISIS-94-002
February, 1994

Katerina Yamalidou, John Moody,
Michael Lemmon and Panos Antsaklis
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556

Interdisciplinary Studies of Intelligent Systems
Some funding for this research was provided by NSF-92-16559 and IRI-91-09298.

Abstract

This report describes a method for constructing a Petri net controller for a discrete event system modeled by a Petri net. The controller consists only of places and arcs and is computed based on the concept of place invariants of the net. The size of the controller is proportional to the number of the constraints which must be satisfied. This method can accommodate constraints written as logic formulas, algebraic inequalities or equalities.

Keywords: Discrete Event Systems, Petri Nets

1 Introduction

Petri nets are a very appropriate tool for the study of discrete-event dynamical systems because of their power and flexibility. In the past they have been used extensively to model and simulate many kinds of systems . Their use in control is somewhat limited and only recently some studies have been conducted towards this direction [1, 2, 4].

Holloway and Krogh [1] used *controlled* Petri nets to control systems that can be modeled as cyclic controlled marked graphs, which is a special class of Petri nets. Their method is valid for cyclic systems modeled by controlled marked graphs. They have shown how to synthesize a maximally permissive feedback controller which guarantees that none of the forbidden states will occur. Their method does not require an exhaustive search of the system's state space, is computationally effective and polynomial in the number of forbidden conditions and in the number of places of the net. The drawback is that it is applicable to a limited class of systems.

Yamalidou [2, 3] formulated the control problem of discrete-event chemical processes as a linear optimization problem based on the Petri net model of the process. The control actions which bring the system from its initial state to a desired final state are computed over a time horizon, while a set of constraints are satisfied and a cost function is minimized. The constraints are written as Boolean expressions which are then transformed into sets of linear inequalities. Since the resulting optimization problem is an integer programming problem, this approach has the drawback that a solution is difficult to obtain for medium and large size systems.

Boissel [4] used simulated annealing to compute a Petri net controller for a discrete-event system modeled by a Petri net. The method can be applied to any system modeled by uncolored Petri nets and can also handle time. Although successful in producing a maximally permissive optimal controller, the method is computationally unattractive for large systems since it involves the construction of the reachability tree several times during the execution of the algorithm.

The method presented here computes a Petri net controller for a discrete-event system modeled by an untimed Petri net and is based on the net's place invariants. The method works for systems whose constraints can be expressed as equalities, inequalities or logic expressions and may involve elements of the marking and/or the firing vector. Constraints that involve the firing vector are transformed, using two different methods, into constraints that involve only the marking vector, allowing the controller to be designed on the principle of place of invariants.

The controller consists of places which are connected to the transitions of the process Petri Net in such a way that it is guaranteed that the system does not enter a forbidden state. The combined process/controller net possesses the necessary place invariants to insure that the set of constraints is not violated.

The controller is not necessarily optimal in size, but it is easily computed and its size is proportional to the number of constraints of the process. Furthermore it can be used as a very good first guess for other methods that are able to compute an optimal Petri net controller, since its relatively small size will reduce the computations needed to reach the optimal controller. The controller is maximally permissive in that it forces the set of constraints to be obeyed, while allowing any action that is not directly or indirectly forbidden by the constraints. If the constraints on a net's performance are written in terms of the firing vector, then there are situations in which the maximal permissive of the control method can only be guaranteed if the net is safe.

This report is structured as follows. First the theory concerning the invariants of the Petri net is discussed briefly. Then the method itself is presented in detail. Different kinds of constraints are discussed. Examples are presented in detail. Finally the conclusions and further research directions are stated.

2 Net Invariants

One of the structural properties of Petri nets, i.e. properties that depend only on the topological structure of the Petri net and not on the net's initial marking, are the net invariants. There are two kinds of invariants: *place invariants* and *transition invariants* [5].

Place invariants are sets of places whose token count remains always constant. They are represented by an n-column vector X, where n is the number of places of the Petri net, whose non-zero entries correspond to the places that belong to the particular place invariant and zeros everywhere else. Every integer vector X which satisfies the following equation

$$\mu^T \cdot X = \mu_0^T \cdot X \tag{1}$$

where μ_0 is the net's initial marking, while μ represents any subsequent marking, defines a place invariant. Equation (1) means that the possibly 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 are defined by all integer vectors which satisfy the following equation

$$X^T \cdot D = 0 \tag{2}$$

where D is the $n \times m$ composite change matrix of the Petri net with n being the number of places and m the number of transitions of the net. It is easily shown that every linear combination of place invariants is also a place invariant for the net.

Transition invariants denote which transitions must fire and how many times each, so that the initial marking is repeated. They are represented by an m-column vector Y which contains integers in the positions corresponding to the transitions belonging to the transition invariant and zeros everywhere else. The integers denote how many times the corresponding transition must fire in order for the initial marking to be repeated. They can be computed from the following equation

$$D \cdot Y = 0 \tag{3}$$

As with place invariants, any linear combination of transition invariants is also a transition invariant for the Petri net. The existence of transition invariants in the Petri net denotes a cyclic behavior.

Place and transition invariants are important means for analyzing Petri nets since they allow for the net's structure to be investigated independently of any dynamic process [6]. Another advantage of the invariants is that analysis can be performed on local subnets without considering the whole system. Invariants are also used for model verification.

3 Description of the Method

The method described in this report requires that the process to be controlled is modeled by a Petri net and constructs a Petri net controller which is attached to the process net. The constraints which must be satisfied by the process can be written as logic expressions, inequalities or equalities. In this section the method will be presented in detail, while the following sections deal with different types of constraints.

Assume that the system to be controlled is modeled by a Petri net with n places and m transitions and it must satisfy the following constraint

$$\mu_i + \mu_j \leq 1 \tag{4}$$

where μ_i and μ_j are the markings of places p_i and p_j of the process net. The above simply means that at the most one of the two places p_i and p_j can be marked, or, in other words, both places cannot be marked at the same time.

This inequality constraint can be transformed into an equality by introducing a *slack* variable μ_s into it. The constraint then becomes

$$\mu_i + \mu_j + \mu_s = 1 \tag{5}$$

The slack variable in this case represents a new place p_s which receives the excess tokens, thus insuring that the sum of tokens in the set of places μ_i and μ_j is always less than or equal to 1. This place belongs to the *controller net*. The structure of this net will be computed by noticing that the introduction of the slack variable introduces a place invariant for the overall system defined by eq. (5). It is obvious that there will be as many controller places as there are constraints of the type (4), so the size of the controller is proportional to the number of constraints of type (4).

Since a new place has been added to the net, the composite change matrix D of the overall *controlled* system is the original $n \times m$ matrix D_p of the system increased by a row corresponding to the place introduced by the slack variable. This new row belongs to the composite change matrix of the controller, called D_c. The arcs connecting the controller place to the original Petri net of the system will be computed by the place invariant equation (2) where the unknowns are the elements of the new row of matrix D while the vector X_i is the place invariant defined by eq. (5). These computations are described below.

First note that the problem can be stated in general, as follows. All constraints of the type of (4) can be grouped and written in matrix form as

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

where μ_p is the marking vector of the Petri net modeling the process, L is an $n_c \times n$ matrix, b is an $n_c \times 1$ vector and n_c is the number of constraints of the type of (4).

Similarly all place invariant equations of the type of (5), generated after the introduction of the slack variables, can be grouped in matrix form as follows

$$L \cdot \mu_p + \mu_c = b \tag{7}$$

where μ_c is an $n_c \times 1$ vector which represents the marking of the controller places.

The place invariant defined by eq. (5) must satisfy the place invariant equation (2). The following matrix equation is the place invariant equation for all invariants defined by (7)

$$X^T \cdot D = [L \;\; I] \cdot \begin{bmatrix} D_p \\ D_c \end{bmatrix} = 0 \quad \Leftrightarrow$$

$$L \cdot D_p + D_c = 0 \quad \Leftrightarrow$$

$$D_c = -L \cdot D_p \tag{8}$$

where I is an $n_c \times n_c$ identity matrix since the coefficients of the slack variables in the constraints are all equal to 1. The matrix D_c contains the arcs that connect the controller places to transitions of the process net. So, given the Petri net model of the process (D_p) and the constraints that the process must satisfy (n_c, L and b), the Petri net controller (D_c) is defined by eq. (8).

The initial marking of the controller Petri net should also be calculated. The initial marking of the controller places μ_{c0} must be such that the place invariant equations (7) are satisfied and depends on the initial marking of the places of the process Petri net which participate in the place invariants. Given eq. (1), eq. (7) can be written for the initial marking vector

$$L \cdot \mu_{p_0} + \mu_{c_0} = b \quad \Leftrightarrow$$

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

As an example consider the simple Petri net of figure 1. The net has three places and four transitions. The composite change matrix of this net is

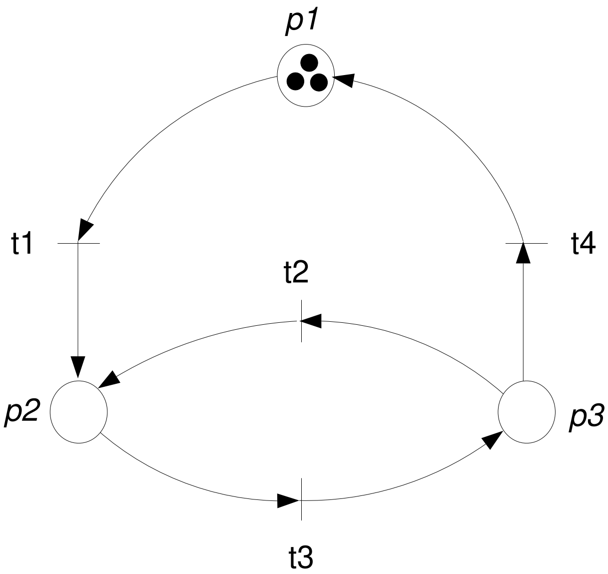

Figure 1: A Simple Cyclic Petri Net.

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

while its initial marking is

$$\mu_{p_0} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \end{bmatrix}$$

D_p is of rank 2, thus it has one place invariant which includes the entire net, i.e., $D_p^T \cdot X = 0$ where $X = [1\ 1\ 1]^T$. The objective is to control the net so that places p_2 and p_3 never contain more than one token, i.e. we wish to enforce the constraint

$$\mu_2 + \mu_3 \leq 1 \tag{10}$$

Using the matrix notation of equation (6) we have

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

$$b = 1$$

The uncontrolled net does not satisfy the desired constraint since $[0\ 1\ 1]^T$ is not a place invariant of the net. A slack variable μ_s is introduced and the inequality (10) becomes an equality

$$\mu_2 + \mu_3 + \mu_s = 1 \tag{11}$$

The slack variable μ_s denotes the marking of the place p_s which belongs to the controller. Equation (11) represents the desired invariant $X = [0\ 1\ 1\ 1]^T$ which will be forced on the controlled Petri net. The composite change matrix of the controller net is computed by equation (8):

$$D_c = -L \cdot D_p = \begin{bmatrix} -1 & 0 & 0 & 1 \end{bmatrix}$$

The initial marking of the controller place is computed from eq. (9):

$$\mu_{s_0} = 1 - L \cdot \mu_{p_0} = 1$$

The structure of the controlled Petri net is then described by the composite change matrix D

$$D = \begin{bmatrix} D_p \\ D_c \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 0 \\ 0 & -1 & 1 & -1 \\ -1 & 0 & 0 & 1 \end{bmatrix}$$

while its initial marking is

$$\mu_0 = \begin{bmatrix} \mu_{p_0} \\ \mu_{s_0} \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

The Petri net graph of the controlled system is shown in figure 2.

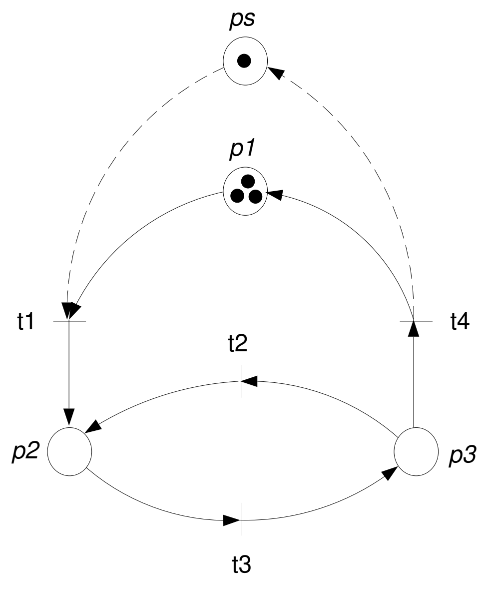

Figure 2: The Controlled Petri Net of the System of Figure 1.

The Petri net of figure 1 is cyclic. In order to show that this method works also for non-cyclic systems, consider the Petri net shown on figure 3. It is a non-cyclic Petri net because of the transition t_5 which has been added as input transition to place p_2. The composite

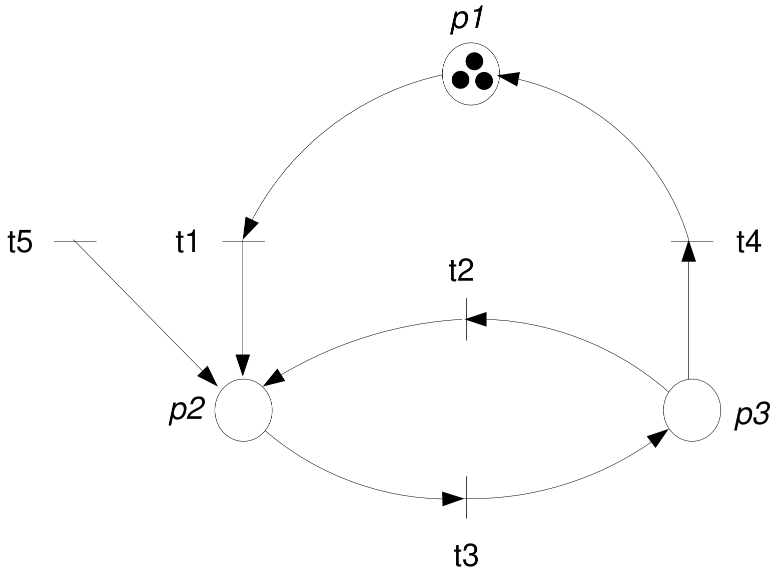

Figure 3: A Simple Non-Cyclic Simple Petri Net.

change matrix that describes this Petri net is

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

The initial marking vector remains unchanged. The rank of D_p is 3 so the uncontrolled net has no place invariants.

The specification is still the one described by eq. (10), i.e. the sum of tokens in places p_2 and p_3 should never exceed 1. As before, a slack variable μ_s makes the constraint an equality as in eq. (11) and corresponds to a controller place p_s. Places p_2, p_3 and p_s form a place invariant and the composite change matrix of the process should be increased by a row, the elements of which are computed as shown below

$$D_c = -L \cdot D_p = \begin{bmatrix} -1 & 0 & 0 & 1 & -1 \end{bmatrix}$$

In addition to the arcs computed before, the controller place is also connected to the transition t_5. The initial marking of the controller place is computed from eq. (9):

$$\mu_{s_0} = 1 - L \cdot \mu_{p_0} = 1$$

The composite change matrix of the controlled Petri net is

$$D = \begin{bmatrix} D_p \\ D_l \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 \\ 1 & 1 & -1 & 0 & 1 \\ 0 & -1 & 1 & -1 & 0 \\ -1 & 0 & 0 & 1 & -1 \end{bmatrix}$$

while its initial marking is

$$\mu_0 = \begin{bmatrix} \mu_{p0} \\ \mu_{s0} \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

The new controlled Petri net is shown in figure 4

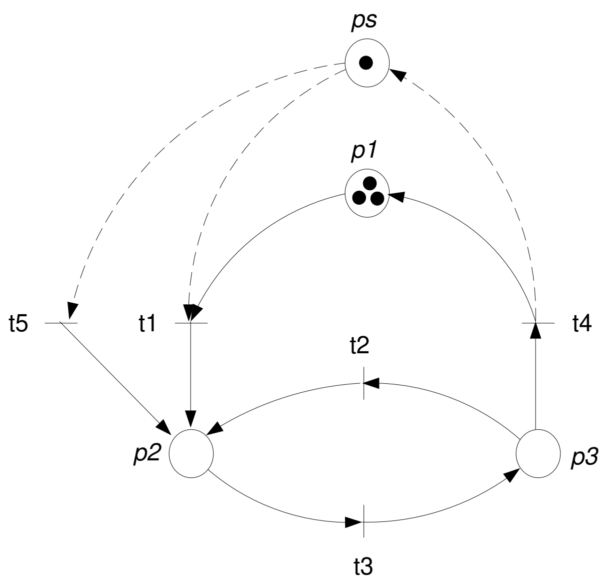

Figure 4: The Controlled Petri Net of Figure 3.

4 Algebraic Transformations

Since the method presented here is based on the concept of place invariants, the constraints must be expressed as a weighted sum of μ's so that the controller can be constructed as discussed in section 3. However not all constraints can be written directly in this form, so all other types of constraints must be first transformed to the form that the method can handle. This section examines different types of constraints and describes appropriate transformations for each of them based on the concept of replacing an element of the firing vector q with the sum of the input places of the corresponding transition. Three general categories are distinguished: logical constraints, inequality constraints and equality constraints. Several cases are considered within each category.

4.1 Logical Constraints

If the constraints can be written as well-formed Boolean formulas of the form

$$A \rightarrow \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_g \tag{12}$$

where A is a propositional variable and each of the Φ_i is an elementary disjunction of the form

$$\Phi_i \equiv \Psi_{i_1} \vee \Psi_{i_2} \vee ... \vee \Psi_{i_{h_i}} \tag{13}$$

and each of the Ψ_{i_j} is a propositional formula, then it has been proved [2] that each of them can be translated to an equivalent set of g simultaneous linear inequalities

$$(1 - \Psi_{i_1}) + (1 - \Psi_{i_2}) + ... + (1 - \Psi_{i_{h_i}}) + A \leq h_i \text{ for } i = 1, 2, ..., g \tag{14}$$

The set of all these inequalities will now represent the constraints that must be satisfied by the system. The system of inequalities should be refined after this transformation in order to eliminate those inequalities that appear more than once.

The inequalities produced from the transformation of (12) can then be handled as described in the appropriate sections below.

4.2 "Less than or Equal to" Constraints

4.2.1 Constraints Containing Marking Vector Elements Only

Assume that the constraints which must be satisfied by the system can be written as a sum of elements of the marking vector

$$\sum_{i=1}^{l} \mu_i \leq k \tag{15}$$

This type of constraints means that the sum of tokens in places $p_1, p_2, ..., p_l$ of the Petri net should never exceed the integer k. This type of constraint is already in the desired form since it is an inequality similar to the one in (4). By introducing a slack variable μ_s into this constraint it can be forced to become an equality

$$\sum_{i=1}^{l} \mu_i + \mu_s = k \tag{16}$$

The method then applies as described in section 3 above. A constraint which contains the weighted sum of μ' is treated in the same way. The coefficients of the μ's will be contained in L.

Note that the constant k in (15) does not play any role in the structure (i.e. the number of places or arcs) of the controller. It defines the constant token count of the invariant described by eq. (16) and should be taken into account when the Petri net is marked initially.

4.2.2 Constraints Containing Both Marking and Firing Vector Elements

Another type of constraints may contain elements of both the marking and the firing vectors. Such constraints link the occurrence of events to part or all of the current system's state vector. They denote that an action can be taken only if the system's state allows it [2].

To illustrate the functionality of such a constraint and the difference between this type of constraints and the ones presented in section 4.2.1, consider a simple process consisting of two valves Valve 1 and Valve 2, represented by v_1 and v_2 respectively, and the pipes that connect them. Assume that Valve 2 must be opened before Valve 1. If the system is modeled by a Petri net as described in [3], where the valve states are modeled as places while transitions model the events of opening and closing the valves, then this requirement can be written as follows

$$\mu_{v_1 open} + q_{openv_2} \leq 1 \tag{17}$$

where $\mu_{v_1 open}$ is the marking of the place $p_{v_1 open}$ and q_{openv_2} if the element of the firing vector which corresponds to the transition t_{openv_2} of the Petri net model of the valve and pipe system. The above constraint means that as long as Valve 1 is open, Valve 2 cannot open, so if both valves must be open, Valve 2 has to be opened first. Compare the above expression with the following

$$\mu_{v_1 open} + \mu_{v_2 open} \leq 1 \tag{18}$$

This constraint, which contains elements of the marking vector only does not allow both valves to be open at the same time. So, constraints that contain elements of the marking vector only govern the state of the system, while the constraints that contain marking and firing vector elements define the events that the system's state allows to occur.

In order for the method described in this report to work, the constraints containing elements of both the marking and the firing vectors must be transformed to constraints containing only elements of the marking vector. Below we show how this is done. We distinguish various cases. Note that some of the transformations produce a maximally permissive controller for *safe* Petri nets only, i.e. for nets whose places can receive at the most one token.

Assume that the constraint concerns one place and one transition only and is of the type

$$\mu_i + q_j \leq 1 \tag{19}$$

This means that the transition t_j cannot fire if place μ_i is marked. In order to transform the above inequality into one that is in the form of (15), we make use of the enabling condition of Petri net theory which states that a transition can fire if and only if all its input places are marked. This implies that q_j can be substituted by the sum of its input places in the left part of (19). The right part should also be modified. The constraint then becomes

$$\mu_i + \sum_{j=1}^{c_j} \mu_j \leq c_j \tag{20}$$

where $\mu_{j_1}, \mu_{j_2}, ..., \mu_{c_j}$ are all the input places of transition t_j and c_j is the number of the input places of transition t_j. The constraint in its new form will not allow more than c_j of the places in the left part of (20) to be marked, so if place p_i is marked not all of the input places of transition t_j may be marked and, consequently, t_j cannot fire. On the other hand, if all of the input places of t_j are marked, then μ_i cannot be marked since this will violate (20). The constraint in its new form (20) is in the form of (15) since it contains only elements of the marking vector. Note that μ_i cannot be an input place of transition t_j since this is not allowed by the initial constraint (19), so it cannot appear twice in (20).

If the constraint concerns one transition only but more than one places, it has the form

$$\sum_{i=1}^{l} \mu_i + q_j \leq 1 \tag{21}$$

If transition t_j has only one input place, then the transformation described above produces an equivalent constraint and eq. (21) becomes

$$\sum_{i=1}^{l} \mu_i + \mu_j \leq 1 \tag{22}$$

where μ_j is the sole input place of t_j. The constraint has now the form of (15). Again, none of the $\mu_1, \mu_2, ..., \mu_l$ can be the input place of q_j since this is not allowed by (21).

If, on the other hand, transition t_j has more than one input place, then a transformation similar to the one for (19) will not produce an equivalent constraint. The constraint in (21) must be replaced by an equivalent set of constraints which lists all the cases not allowed by (21) and are in a form that can be transformed to (15). This set will have the form

$$\text{Part 1} \quad \left\{ \sum_{i=1}^{l} \mu_i \leq 1 \right.$$

$$\text{Part 2} \quad \left\{ \begin{array}{ccccc} \mu_1 & + & q_j & \leq & 1 \\ \mu_2 & + & q_j & \leq & 1 \\ & & \vdots & & \\ \mu_l & + & q_j & \leq & 1 \end{array} \right. \tag{23}$$

The inequality in the first part of the set is already in the form of (15). Each of the inequalities of the second part of the group involving μ_i's and q_j has the form of (19) and can be brought to the form of (20) as shown previously.

The next case to be considered is a more general form of (21)

$$\sum_{i=1}^{l} \mu_i + q_j \leq l \tag{24}$$

Note that the number of μ's involved in the constraint is equal to the constant of the right side l. Following the same reasoning as above, q_j is replaced by the set of its input places, the right side of the inequality changes accordingly and (24) becomes

$$\sum_{i=1}^{l} \mu_i + \sum_{j=1}^{c_j} \mu_j \leq l + c_j - 1 \tag{25}$$

where c_j is the number of input places of t_j. Again, the transformed constraint (25) is in the form of (15). Contrary to the above cases, some of the input places of q_j may be involved in the constraint of (24). This is taken into account in (25) since they are counted twice in the right side, in both l and c_j.

Another case in this category is the following

$$\sum_{i=1}^{l} \mu_i + q_j \leq k \tag{26}$$

where $k < l$. If transition t_j has only one input place, then the above expression can be transformed into

$$\sum_{i=1}^{l} \mu_i + \mu_j \leq k \tag{27}$$

If t_j has more than one places, this type of constraint has to be replaced by a set of constraints listing all possible combinations, as in the case of the constraints in eq. (21). The number of terms in the left part of the inequalities containing the term q_j has to be $k+1$. Then these inequalities will be in the form of eq. (24), while the inequality which concerns $\sum_{i=1}^{l} \mu_i$ will be in the form of eq. (15).

Lastly the constraint may have the more general form

$$\sum_{i=1}^{l} \mu_i + \sum_{j=1}^{g} q_j \leq k \tag{28}$$

where $k < l + g$. This type of constraints cannot be transformed as it is in general. It must be first replaced by an equivalent set of constraints, i.e. a set of inequalities which describe all the possibilities contained in eq. (28), and will be in one of the forms described in this section. Each one of the constraints of the set should then be transformed appropriately.

4.2.3 Constraints Involving Firing Vector Elements Only

These constraints refer to the simultaneous occurrence of two or more events. All the possible cases are considered.

Assume that the constraint is of the type

$$\sum_{j=1}^{l} q_j \leq l - 1 \tag{29}$$

which indicates that not all transitions $q_1, q_2, \ldots, q_l$ can fire simultaneously.

Again, the enabling condition of Petri net theory suggests that the constraint can be transformed into an equivalent constraint by substituting for each transition the sum of its input places in the left part of the inequality (29) and by modifying the right part conformably. The constraint then becomes

$$\sum_{j=1}^{l} \mu_{j_1} + \mu_{j_2} + ... + \mu_{c_j} \leq \sum_{j-1}^{l} c_j - 1 \tag{30}$$

where $\mu_{j_1}, \mu_{j_2}, ..., \mu_{c_j}$ are the input places of transition t_j, while c_j denotes the number of the input places of transition t_j. The transformed constraint does not allow all of the input places of all transitions to be marked simultaneously, so it ensures that not all transitions can fire together. The transformed constraint contains only elements of the marking vector, so it falls into one of the categories described in section 4.2.1.

Any other type of constraint containing only elements of the firing vector has the form

$$\sum_{j=1}^{l} q_j \leq k \tag{31}$$

where $k \leq l - 2$. This must be first replaced by an equivalent set of inequalities of the type of eq. (29), listing all the possibilities, before it can be transformed further.

4.3 “Greater than or Equal” Constraints

The transformations described below are valid for safe Petri nets only, i.e. for Petri nets whose places can have at the most one token.

4.3.1 Constraints Containing Marking Vector Elements Only

In some cases it is necessary to say that a set of places contains at least k tokens. This is expressed by the following constraint

$$\sum_{i=1}^{l} \mu_i \geq k \tag{32}$$

This means that at least k of the l places must be marked. This constraint can become an equality if we add an *excess* variable μ_e to the left part of it

$$\sum_{i=1}^{l} \mu_i - \mu_e = k \tag{33}$$

As in the case of the slack variable, the excess variable μ_e introduces a place which belongs to the controller and forces an invariant formed by places $\mu_1, \mu_2, ..., \mu_l$ and μ_e in the controlled Petri net. As shown by eq. (33) it is the *weighted* sum of the tokens in the places of the invariant that remains constant. Since the constraint has been brought to the form of an invariant it can be treated similarly to eq. (16), the only difference being that the coefficient of the slack variables is -1 now. The equation for computing the matrix of the controller net is

$$[L \quad -I] \cdot \begin{bmatrix} D_p \\ D_c \end{bmatrix} = 0 \quad \Leftrightarrow$$

$$L \cdot D_p - D_c = 0 \quad \Leftrightarrow$$

$$D_c = L \cdot D_p \tag{34}$$

while the initial marking vector of the controller places is

$$L \cdot \mu_{p0} - \mu_{c0} = b \quad \Leftrightarrow$$

$$\mu_{c0} = L \cdot \mu_{p0} - b \tag{35}$$

4.3.2 Constraints Containing Both Marking and Firing Vector Elements

The first case considered is the constraint containing one element of μ and one element of q

$$\mu_i + q_j \geq 1 \tag{36}$$

The above expression means that whenever μ_i is not marked, t_j should fire and whenever t_j does not fire μ_i should be marked. As done with all kinds of constraints so far, it is necessary to transform this constraint to an expression containing elements of the marking vector only. In order to do this, we must first express the constraint as a well-formed Boolean formula with same meaning. Eq. (36) can be replaced by the following logical expression, since they both contain the same meaning

$$\neg\mu_i \rightarrow q_j \tag{37}$$

The above simply means that whenever μ_i is not true (i.e. p_i is not marked) q_j must be true (i.e. t_j must fire) and whenever q_j is not true (t_j does not fire) μ_i must be true (i.e. p_i must be marked). According to the Petri net theory a transition can fire if and only if all its input places are marked. This allows us to replace q_j in (37) with the conjunction of all its input places. It then becomes

$$\neg\mu_i \rightarrow \mu_{j1} \wedge \mu_{j2} \wedge \ldots \wedge \mu_{c_j} \tag{38}$$

where c_j is the number of input places of t_j. The above contains only elements of the marking vector. It is a well-formed formula of the type (12) and, as shown in section 4.1, it is equivalent to and can be replaced by the following set of inequalities

$$\begin{array}{c} (1-\mu_{j_1}) + (1-\mu_i) \leq 1 \\ (1-\mu_{j_2}) + (1-\mu_i) \leq 1 \\ \vdots \\ (1-\mu_{c_j}) + (1-\mu_i) \leq 1 \end{array} \tag{39}$$

The term $\neg\mu_i$ was replaced by $1-\mu_i$ because the net is safe. This substitution holds because $\neg\mu_i$ and $1-\mu_i$ have the same value as shown below

$$\begin{array}{llll} \text{for } \mu_i = 1 & \neg\mu_i \equiv 0 & \text{and} & 1-\mu_i = 0 \\ \text{for } \mu_i = 0 & \neg\mu_i \equiv 1 & \text{and} & 1-\mu_i = 1 \end{array} \tag{40}$$

Further manipulation of the inequalities in (39) gives

$$\begin{array}{c} \mu_{j_1} + \mu_i \geq 1 \\ \mu_{j_2} + \mu_i \geq 1 \\ \vdots \\ \mu_{c_j} + \mu_i \geq 1 \end{array} \tag{41}$$

All the inequalities in (41) have the form of (32) and can be dealt with accordingly.

The general form of the constraints in this category is

$$\sum_{i=1}^{l} \mu_i + \sum_{j=1}^{g} q_j \geq k \tag{42}$$

Following the reasoning for the case of eqn. (36), the above constraint must first be written as a logical expression with the same meaning. Then each of the parts of the logical expression must be transformed as shown above. Care should be taken to simplify the set of well-formed Boolean formulas in order to avoid redundancy.

4.3.3 Constraints Containing Firing Vector Elements Only

The last case in this section is the case when the constraint contains elements of the firing vector only. The general form is

$$\sum_{j=1}^{g} q_j \geq k \tag{43}$$

This implies that at every firing instant at least k of the g transitions $t_1, t_2, ..., t_g$ must fire. These constraints must be first replaced by logical expressions having the same meaning, as in section 4.3.2, before any further transformation is possible. As an example consider the simpler case

$$q_1 + q_2 \geq 1 \tag{44}$$

This means that at any given firing instant one of the two transitions t_1 and t_2 must fire. Logically this can be expressed as

$$\neg q_1 \rightarrow q_2 \tag{45}$$

Each of the q's in (45) can be substituted by the conjunction of the markings of its input places. The above then becomes

$$\neg[\mu_{1_1} \wedge \mu_{1_2} \wedge ... \wedge \mu_{c_1}] \rightarrow \mu_{2_1} \wedge \mu_{2_2} \wedge ... \wedge \mu_{c_2} \tag{46}$$

where $\mu_{1_1}, \mu_{1_2}, ..., \mu_{c_1}$ and $\mu_{2_1}, \mu_{2_2}, ..., \mu_{c_2}$ are the sets of input places of transitions t_1 and t_2 respectively. The expression in (46) can be written as

$$\begin{array}{ll} (\neg\mu_{1_1} \rightarrow \mu_{2_1} \wedge \mu_{2_2} \wedge ... \wedge \mu_{c_2}) & \wedge \\ (\neg\mu_{1_2} \rightarrow \mu_{2_1} \wedge \mu_{2_2} \wedge ... \wedge \mu_{c_2}) & \wedge \\ \vdots & \\ (\neg\mu_{c_1} \rightarrow \mu_{2_1} \wedge \mu_{2_2} \wedge ... \wedge \mu_{c_2}) & \end{array} \tag{47}$$

Each of the above has the form of (12) and each can be substituted by a set of inequalities as shown in section 4.1.

4.4 Equality Constraints

The last general case that remains to be considered is the case where the constraints are written as plain equalities. We distinguish the different subcategories.

4.4.1 Equality Constraints Containing Marking Vector Elements Only

These constraints are written as

$$\sum_{i=1}^{l} \mu_i = k \tag{48}$$

The above equation means that places $p_1, p_2, ..., p_l$ must always form a place invariant. This is really a specification for the system and should have been incorporated into the Petri net model. If this invariant is not already part of the Petri net model, it should become at this point. This is done by modifying the composite change matrix D_p of the Petri net so that eq. (2) holds, where X contains the invariant defined by eq. (48). The new elements of D_p represent the arcs which should be added to the Petri net so that the place invariant becomes part of it.

4.4.2 Equality Constraints Containing Both Marking and Firing Vector Elements

Assume that the constraint is of the type

$$\sum_{i=1}^{k} \mu_i + \sum_{j=1}^{l} q_j = k + l \tag{49}$$

This constraint means that at all times all of $\mu_1, \mu_2, ..., \mu_k$ should be marked and all of $q_1, q_2, ..., q_l$ should fire. It is rare that such a requirement is imposed on a system, but it may occur. The constraint can be transformed by substituting each of q_j with the sum of its input places and by modifying the right part accordingly. The transformed constraint is

$$\sum_{i=1}^{k} \mu_i + \sum_{j=1}^{l} \sum_{s=1}^{c_j} \mu_{j_s} = k + \sum_{j=1}^{l} c_j \tag{50}$$

where c_j is the number of input places of transition t_j. The constraint has now the form of eq. (48).

4.4.3 Equality Constraints Containing Firing Vector Elements Only

Some equality constraints may contain only elements of the firing vector. A simple example is

$$q_i + q_j = 1 \tag{51}$$

The above means that one of the two transitions should fire at every instant. In other words, if q_i will not fire next then q_j must fire and vice versa. This can be written as a logical formula

$$(q_i \rightarrow \neg q_j) \wedge (\neg q_i \rightarrow q_j) \tag{52}$$

The same concept can be expressed by means of the input places of the two transitions

$$\begin{aligned} &\left([\mu_{i_1} \wedge \mu_{i_2} \wedge \ldots \wedge \mu_{c_i}] \rightarrow \neg \left[\mu_{j_1} \wedge \mu_{j_2} \wedge \ldots \wedge \mu_{c_j}\right]\right) \wedge \\ &\left(\neg [\mu_{i_1} \wedge \mu_{i_2} \wedge \ldots \wedge \mu_{c_i}] \rightarrow \mu_{j_1} \wedge \mu_{j_2} \wedge \ldots \wedge \mu_{c_j}\right) \end{aligned} \tag{53}$$

These formulas should be separated and brought to the form of eq. (12) and then replaced by inequalities. All constraints of this type should be analyzed as shown above.

5 Transformations of the Petri Net

Another way of transforming constraints which contain elements of the firing vector is described in this section. It is based on a transformation performed on the Petri net model itself.

Assume that a system modeled by a Petri net must satisfy the constraint

$$\mu_i + q_j \leq 1 \tag{54}$$

This means that transition t_j cannot fire if place p_i is marked and vice versa. In order to bring this constraint to a form which contains elements of the marking vector only, the following transformation is done to the process Petri net. Transition t_j is replaced by two transitions and a place between them, as shown in figure 5.

The composite change matrix D_p of the process Petri net is increased by one column and one row since the overall number of transitions and places of the Petri net has each been

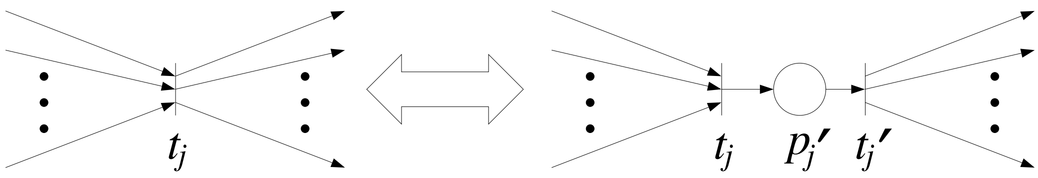

Figure 5: Transformation of a Transition.

increased by one. This transformation is artificial and does not add to or subtract anything from the Petri net model of the process. Its sole purpose is to introduce the place p_j' which records the firing of the transition t_j.

The marking μ_j' of the place p_j' replaces q_j in the constraint (54), which becomes

$$\mu_i + \mu_j' \leq 1 \tag{55}$$

The constraint now contains only μ's and the controller can be computed as described in section 3. Since the method produces a controller consisting of places and arcs only, no part of the controller is connected directly to the place p_j' of the transformation. After the controller structure is computed by the method, the two transitions and the place of the transformation collapse to the original transition.

The same transformation is done to all the transitions which appear in the constraints. Those constraints that contain only q's are treated in the same way.

As an example consider again the Petri net of figure 1. Assume that the constraint that must be satisfied this time is

$$q_2 + q_3 \leq 1 \tag{56}$$

which means that transitions q_2 and q_3 cannot fire at the same time. Each of the two transitions is replaced by two transitions and a place in between. The transformed net is shown in figure 6.

The composite change matrix of the process Petri net after the transformation is

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

and its initial marking is

$$\mu_{p0} = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

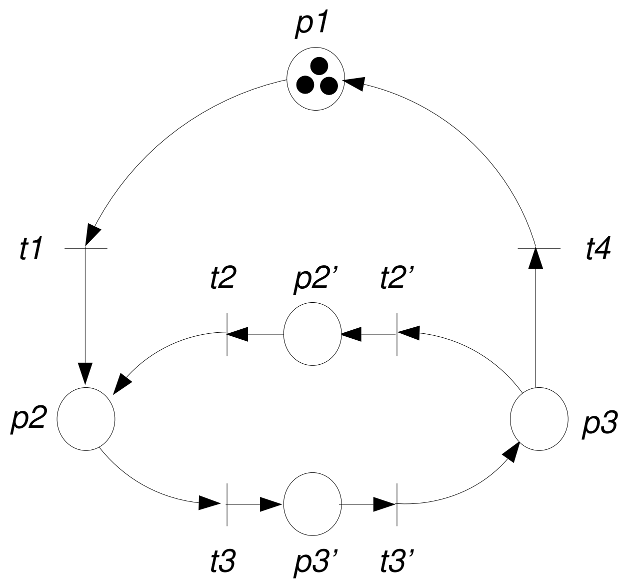

Figure 6: The Transformed Petri Net of Figure 1.

The constraint can now be expressed in terms of the two places ${p_2}'$ and of ${p_3}'$ as

$$ {\mu_2}' + {\mu_3}' \leq 1 \tag{57} $$

The constraint can be written in the matrix form of eq. (6) with

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

The constraint can be converted into an equality by introducing a slack variable μ_s

$$ {\mu_2}' + {\mu_3}' + \mu_s = 1 \tag{58} $$

The slack variable μ_s corresponds to the controller place p_s. The controller structure is computed according to eq. (8)

$$ D_c = -L \cdot D_p = \begin{bmatrix} 0 & -1 & 1 & -1 & 1 & 0 \end{bmatrix} $$

The initial marking of the controller place is computed from eq. (9):

$$ \mu_{s_0} = 1 - L \cdot \mu_{p_0} = 1 $$

The structure of the controlled transformed Petri net is then described by the composite change matrix D

$$ D = \begin{bmatrix} D_p \\ D_c \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & -1 & 1 & 0 \end{bmatrix} $$

while its initial marking is

$$\mu_0 = \begin{bmatrix} \mu_{p0} \\ \mu_{s0} \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

and the controlled transformed Petri net is shown in figure 7

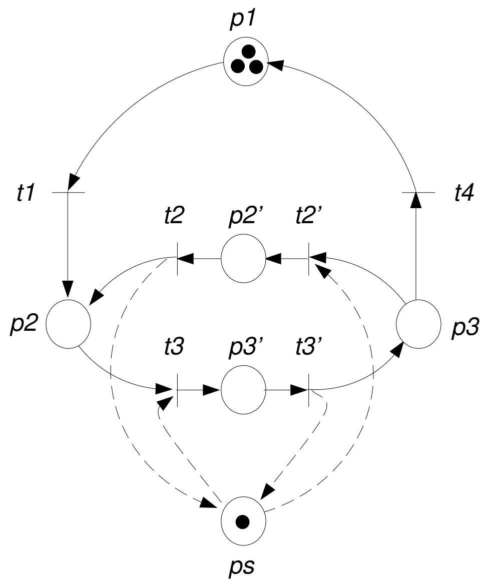

Figure 7: The Controlled Transformed Petri Net of Figure 1.

Once the controller is computed, the two transformations that were introduced collapse to produce the original transitions t_2 and t_3. The second and third column of the matrix D are added to produce the column corresponding to transition t_2 while the third and fourth column of matrix D are added to produce the column corresponding to transition t_3. Rows 4 and 5 corresponding to the two places of the transformation are deleted. So, the structure of the original controlled Petri net is given by

$$D = \begin{bmatrix} D_p \\ D_c \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 0 \\ 0 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

and its initial marking is

$$\mu_0 = \begin{bmatrix} \mu_{p0} \\ \mu_{s0} \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Figure 8 shows the final controlled system of figure 1 with the constraint (56).

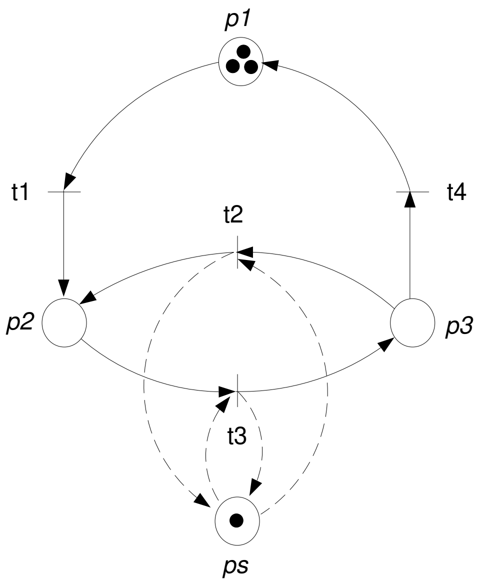

Figure 8: The Controlled Petri Net of Figure 1.

Note that this way of transforming constraints containing q's produces a controller which is connected directly to the corresponding transitions and does not allow these transitions to fire, even if they are enabled, when their firing violates the constraint. This requires that the transitions be controllable. If they are not, then the computation of the controller based on the net transformation is not valid and the constraints must be transformed using the method described in section 4 above.

6 Examples

Two examples illustrating the method described above are presented. The first is the well-known "Cat and Mouse" problem. The second is a flexible manufacturing system used by Holloway and Krogh [1].

6.1 The Cat and Mouse Problem

The "cat and mouse" problem, introduced by Wonham and Ramadge [7], is a popular example in the field of discrete event system control. The problem involves a maze of five rooms where a cat and a mouse can circulate. The rooms are connected with doors through which the animals can pass as shown in figure 9. The problem is to control the doors so that the cat and the mouse can never be in the same room at the same time. The controller should be maximally permissive in the sense that it should grant maximum freedom of movement to both the cat and the mouse. The simple Petri net model of the cat and mouse problem is taken out of Boissel [4] and is shown in figure 10. The upper net concerns the cat while the

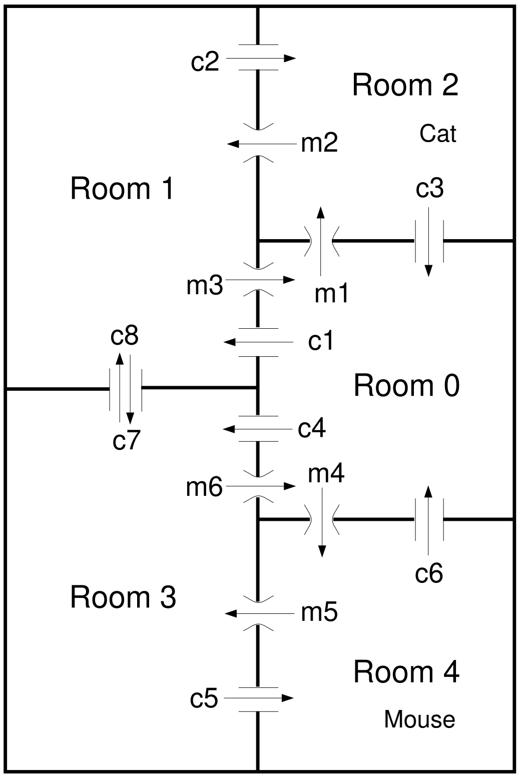

Figure 9: The Cat and Mouse Maze.

lower net concerns the mouse. Each net has five places which model the five rooms of the maze. The transitions model the ability of each animal to pass from one room to the other, according to figure 9. The composite change matrix of the Petri net model for this system is

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

There is one token only in each of the nets since there is one cat and one mouse. The presence of a token in a place indicates that the animal modeled by the token is in the room modeled by the particular place. Initially the cat is in room 2 and the mouse is in room 4 and this is reflected on the initial marking of the Petri net which is shown below

$$\mu_{p0} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_{10} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

D_p is of rank 8, thus it has two place invariants. The first consists of the all five places of the upper net and the second consists of all five places of the lower net.

First it is assume that all the doors of the maze are controllable. The control goal is to insure that the cat and mouse are never in the same room simultaneously. This means that each pair of places, one from the upper net and one from the lower net, that model the same room must contain at the most one token at all times. This requirement is translated into the following five constraints, one for each room

$$\begin{aligned} \mu_1 + \mu_6 &\leq 1 \\ \mu_2 + \mu_7 &\leq 1 \\ \mu_3 + \mu_8 &\leq 1 \\ \mu_4 + \mu_9 &\leq 1 \\ \mu_5 + \mu_{10} &\leq 1 \end{aligned} \tag{59}$$

Since all the above constraints contain only elements of the marking vector, they are already in the form of eq. (15). They can be forced into equalities directly by introducing slack variables, one for each inequality. They become

$$\begin{aligned} \mu_1 + \mu_6 + \mu_{s_1} &= 1 \\ \mu_2 + \mu_7 + \mu_{s_2} &= 1 \\ \mu_3 + \mu_8 + \mu_{s_3} &= 1 \\ \mu_4 + \mu_9 + \mu_{s_4} &= 1 \\ \mu_5 + \mu_{10} + \mu_{s_5} &= 1 \end{aligned} \tag{60}$$

The five slack variables $\mu_{s_1}, \mu_{s_2}, \mu_{s_3}, \mu_{s_4}$ and μ_{s_5} correspond to five new places which belong to the controller.

Using the matrix notation of equation (7) we have

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

The portion of the D matrix associated with the controller is given by eq. (8)

$$\begin{aligned} D_c &= -L \cdot D_p \\ &= \begin{bmatrix} 1 & 0 & -1 & 1 & 0 & -1 & 0 & 0 & 1 & 0 & -1 & 1 & 0 & -1 \\ -1 & 1 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \end{bmatrix} \end{aligned} \tag{61}$$

The initial marking of the slack places is given by eq. (9):

$$\mu_{s0} = 1 - L \cdot \mu_{p0} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

The composite change matrix and initial marking of the controlled net is

$$D = \begin{bmatrix} D_p \\ D_c \end{bmatrix} \qquad \mu_0 = \begin{bmatrix} \mu_{p0} \\ \mu_{s0} \end{bmatrix}$$

The new net prevents the cat and mouse from ever entering the same room, but allows all other legal moves. The controlled net is shown in figure 2 with the controlling arcs shown as dashed lines to distinguish them from the arcs of the original uncontrolled net.

Another problem arises if door c_7 is uncontrollable. The structure and marking of the process Petri net are the same as above, but the constraints must be written differently to assure liveness of the controlled system. The set of constraints to be satisfied now is given below

$$\begin{aligned} \mu_1 + \mu_2 + \mu_4 + \mu_6 + \mu_9 &\leq 1 \\ \mu_3 + \mu_8 &\leq 1 \\ \mu_5 + \mu_{10} &\leq 1 \end{aligned} \tag{62}$$

The corresponding L and b are

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

The controller has three places and the controller matrix is computed as before. The resulting D_c is

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

and the controller's initial marking is

$$\mu_c = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

The controlled Petri net for this case is shown in figure 12.

6.2 Automated Guided Vehicle Coordination

This example, concerning a flexible manufacturing cell has been used by Holloway and Krogh [1]. It includes three workstations, two part-receiving stations and one completed parts station. There are five automated guided vehicles (AGV's) which can transport material between the stations. The routes of the vehicles cross on the floor of the plant and, consequently, there are zones in which two vehicles could be present at the same time. This is a forbidden situation.

The Petri net model of this system, taken out of [1], is shown in figure 13. The shaded areas represent the dangerous zones in which the vehicles' trajectories cross. The vehicles and the parts are modeled by tokens. The marking of the Petri net corresponds to the actual state of the system. A maximally permissive controller, which ensures that at the most one vehicle is present in each zone at any time, should be designed.

The constraints that the process must satisfy concern the presence of the vehicles in the dangerous zones and are expressed by the following inequalities

$$\begin{aligned} \sum_{i \in Z_1} \mu_i &\leq 1 \\ \sum_{i \in Z_2} \mu_i &\leq 1 \\ \sum_{i \in Z_3} \mu_i &\leq 1 \\ \sum_{i \in Z_4} \mu_i &\leq 1 \end{aligned} \tag{63}$$

where Z_j is the set of indices of places which make up zone j. Slack variables are introduced and the inequalities become equalities

$$\sum_{i \in Z_1} \mu_i + \mu_{s_1} = 1$$

$$\begin{aligned}
\sum_{i \in Z_2} \mu_i + \mu_{s_2} &= 1 \\
\sum_{i \in Z_3} \mu_i + \mu_{s_3} &= 1 \\
\sum_{i \in Z_4} \mu_i + \mu_{s_4} &= 1
\end{aligned} \tag{64}$$

The four slack variables define four places for the controller. Each place controls the access of a zone. The composite change matrix of the process is increased by four rows which correspond to the four controller places and constitute the composite change matrix D_c of the controller net. After computing D_c and μ_{c_0} from eqs (8) and (9), the appropriate arcs are added to connect the controller places to the appropriate transitions of the Petri net of the process. The controlled Petri net is shown in figure 14.

7 Conclusions

This paper has presented a particularly simple method for constructing feedback controllers for untimed Petri nets. The method is based on the idea that specifications representing desired plant behaviors can be enforced by making them invariants of the controlled net. In this paper, therefore, a technique was derived which used place invariants representative of logical design specifications. The resulting controller consists only of places and arcs and its size is proportional to the number of constraints. The method can accommodate both controllable and uncontrollable transitions.

The significance of this particular approach to Petri net controller design is that the desired control net can be computed very efficiently by a single matrix multiplication. The resulting controlled system will generally not be optimal in terms of minimizing the number of its places in the controller net. However, the approach yields controllers whose size grows in a polynomial manner with the number of specifications and due to the ease of computation can represent a good initial point in subsequent controller optimization. Consequently, the proposed approach appears to possess significant potential for helping in the design of feedback controllers for a relatively large class of Petri nets.

There are several areas in which the control method based on the place invariants is open for further research. First the method should be expanded to deal with timed Petri nets as these networks are being used more and more often due to their added modeling power. Another important future research goal is a systematic method for transforming a set of constraints into an equivalent set when transitions in the process net are uncontrollable or unobservable. This transformation would be similar to the "supremal controllable sublanguage" used by Ramadge and Wonham in their work on discrete event system control [7, 8]. There are two surmountable problems with the algebraic transformations on constraints that involve the firing vector as presented in section 4. It is possible that a single constraint may be transformed into a large set of subconstraints, and maximal permissiveness of the controller is only guaranteed if the Petri net is safe. It is believed that future study will alleviate both of these difficulties.

References

[1] Holloway L.E. and B.H. Krogh "Synthesis of Feedback Logic for a Class of Controlled Petri Nets", *IEEE Trans. Automat. Contr.*, **35**, 5, pp. 514-523, 1990.

[2] Yamalidou E. and J.C. Kantor, "Modeling and Optimal Control of Discrete-Event Chemical Processes Using Petri Nets", *Comp. Chem. Engng.*, **15**, 7, pp. 503-519, 1991.

[3] Yamalidou E., "Modeling, Optimization and Control of Discrete-Event Chemical Processes Using Petri Net Theory", *Ph.D. Dissertation*, University of Notre Dame, Notre Dame, Indiana, 1991.

[4] Boissel O., "Optimal Feedback Control Design for Discrete-Event Process Systems Using Simulated Annealing", *M.S. Thesis*, University of Notre Dame, Notre Dame, Indiana, 1988.

[5] Reisig W., "Petri Nets", Springer-Verlag, 1985.

[6] Lautenbach O., "Linear Algebraic Techniques for Place/Transition Nets", *Lecture Notes in Computer Science*, vol 254, pp 142-167, 1987.

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

[8] Ramadge, P. J. G., Wonham, W. M., "The Control of Discrete Event Systems," *Proceedings of the IEEE*, vol 77, No. 1, pp 81–97, January 1989.

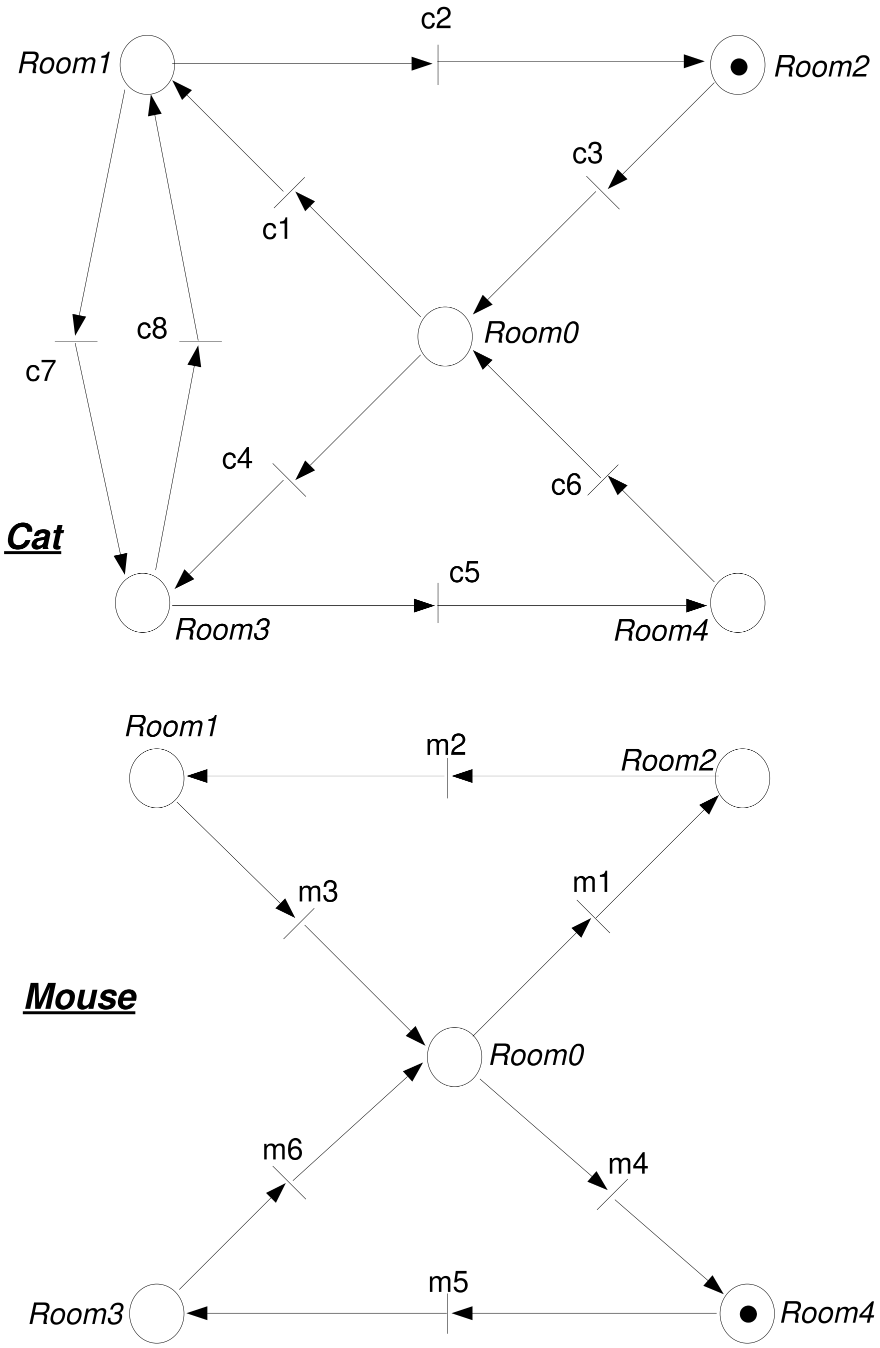

Figure 10: The Petri Net Model of the Cat and Mouse Problem.

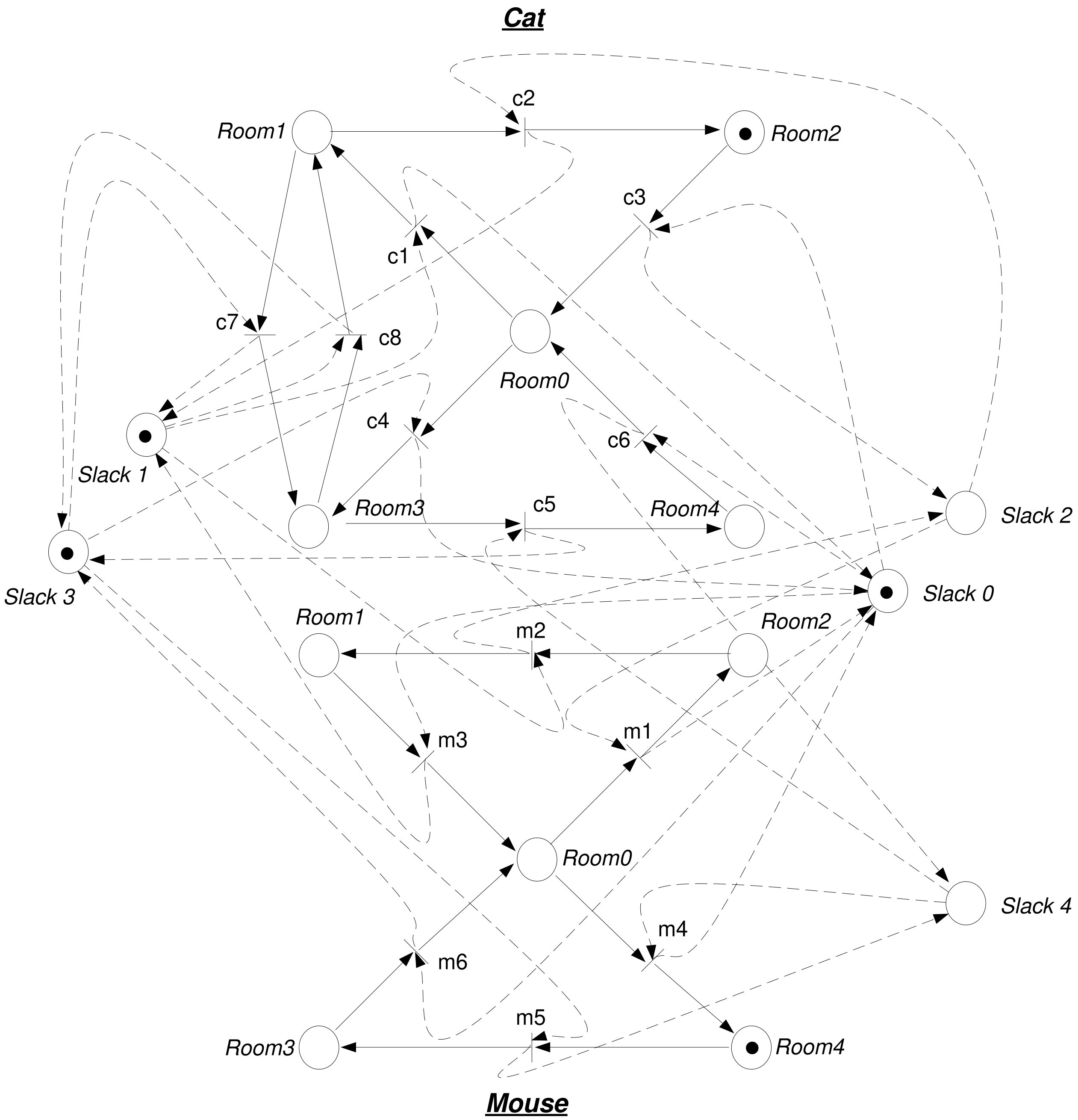

Figure 11: The Controlled Petri Net of the Cat and Mouse Problem.

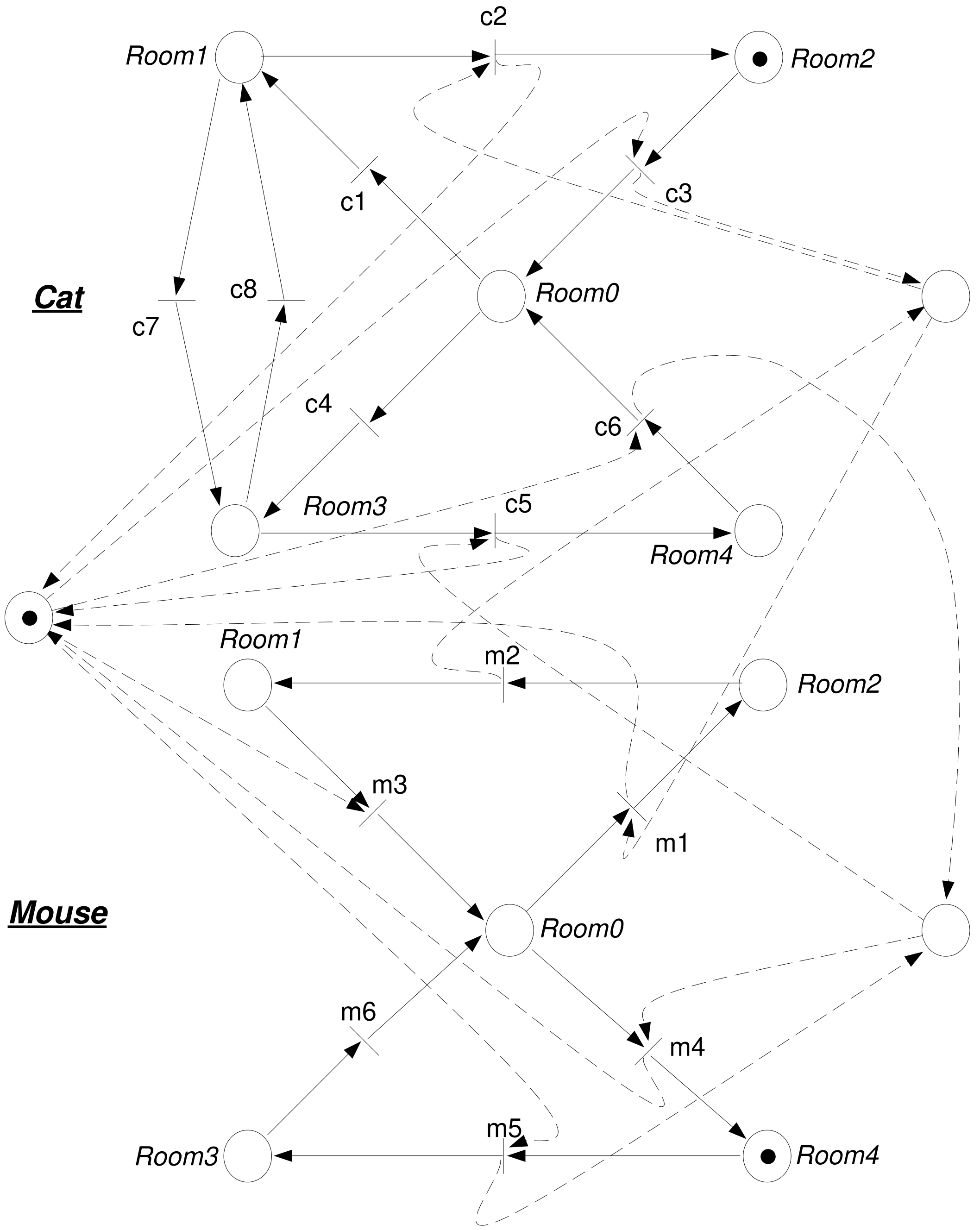

Figure 12: The Controlled Petri Net of the Cat and Mouse Problem.

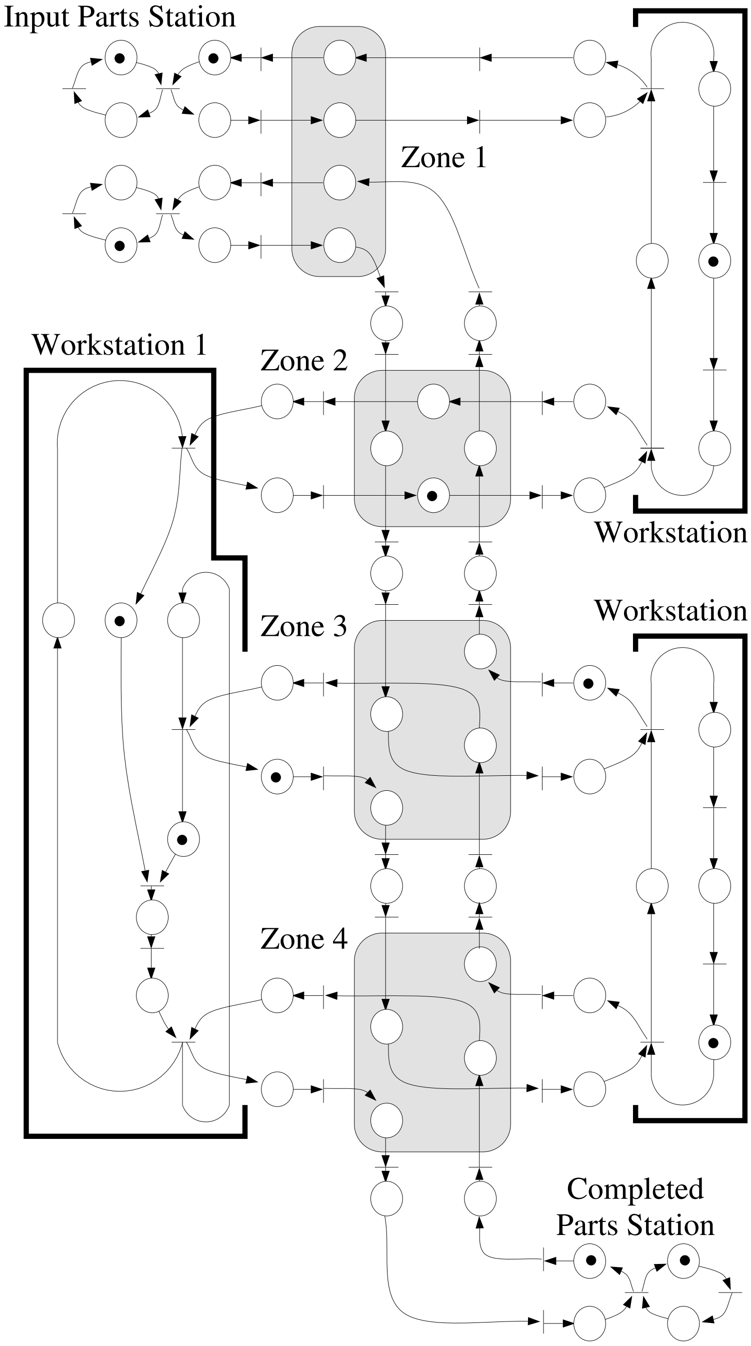

Figure 13: The Automated Guided Vehicle Petri Net.

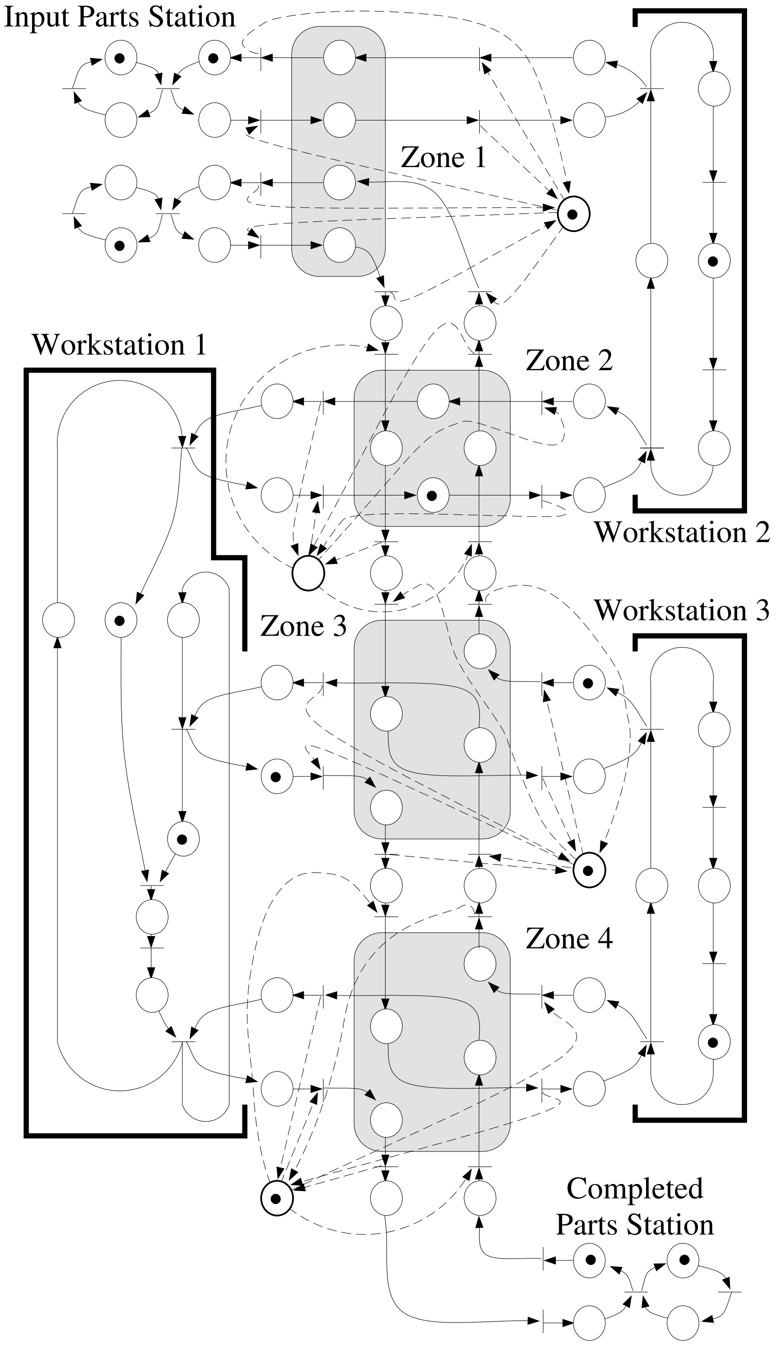

Figure 14: The Controlled AGV Net.