43rd IEEE Conference on Decision and Control
December 14-17, 2004
Atlantis, Paradise Island, Bahamas

Fault Detection and Identification in Petri Net Controllers

Lingxi Li, Christoforos N. Hadjicostis and R. S. Sreenivas

Abstract— **This paper proposes a methodology for providing fault tolerance to Petri net controllers. In order to provide tolerance against faults that may compromise the functionality of the Petri net controller, we construct a redundant Petri net embedding that allows the detection and identification of place and transition faults. A place fault results in an incorrect token-load of a place, and a transition fault arises when the token-load of either the input or output place-set of a transition is not appropriately updated following the firing of a transition. The resulting Petri net controller implementations use redundant places, connections and tokens to impose invariant conditions that allow the detection and identification of faults via *linear parity checks*. The proposed methodology is attractive because the redundant Petri net controller makes efficient use of redundancy and allows for the systematic detection and identification of faults. More specifically, the use of d redundant places enables the detection and identification of up to $d-1$ transition faults and up to $\lfloor d/2 \rfloor$ place faults.**

I. INTRODUCTION

Petri nets have been widely used to model and analyze discrete event dynamic systems. Petri net models can be more compact than automata-based models for the representation of the same system behavior, and the graphical representation of a plant as a Petri net model can have advantages when trying to design a plant controller or supervisor. In order to ensure that a given Petri net plant behaves in a way that meets certain desirable criteria, the authors of [1] studied how a class of generalized mutual exclusion constraints can be enforced via properly constructed monitors. In [2] it was shown that mutual exclusion constraints for acyclic Petri nets with uncontrollable transitions can be reduced to the online solution of an integer linear program.[1] In this paper we propose a fault-tolerant design for controllers that are themselves Petri nets. For instance, the controllers in [3] are Petri nets which are chosen in a way that forces the plant to satisfy certain constraints (control objectives). The idea in [3] is based on enforcing place invariants on the overall Petri net that consists of both the controller and the plant.

This material is based upon work supported in part by the National Science Foundation under NSF ITR Award 0085917 and NSF EPNES Award 0224729. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of NSF.

The authors are with the Coordinated Science Laboratory and the Departments of Electrical and Computer Engineering, and General Engineering, University of Illinois at Urbana-Champaign, USA.

Corresponding author: C. N. Hadjicostis, 357 Coordinated Science Laboratory, 1308 West Main Street, Urbana, IL 61801–2307, USA. Email:chadjic@uiuc.edu.

[1]The mutual exclusion problem for general Petri nets is more complex than ILP-formulations.

Our objective in this paper is to build a *redundant controller* that allows an external checker to detect and identify faults that take place in the controller. More specifically, we are interested in controller faults that lead to an incorrect token-load of a place (place fault) or cause the token-load of either the input or output place-set of a transition not to be properly updated following the firing of a transition (transition fault). Our approach is based on embedding the original controller into a *separate* redundant controller in a way that preserves the state and properties of the original Petri net controller, while enabling the development of systematic ways to detect and identify faults in the redundant Petri net controller. As a result, by performing *linear parity checks* on the combined marking of the original controller places and the additional (redundant) places, our methodology is able to detect and identify faults in the redundant Petri net controller in a systematic manner.

An outline of how the separate fault-tolerant Petri net controller (also referred to as the separate redundant Petri net controller) achieves fault-tolerant control is shown in Fig.1: given a Petri net plant, the original Petri net controller can be obtained based on any method (e.g., a place invariant enforcing controller as in [3]). In order to protect the controller against faults, we add redundant places in a way that does not inhibit any transitions that would otherwise be enabled to fire by the original controller (i.e., the redundant places retain the maximal permissiveness of the original controller). As we will show, these redundant places can be added systematically so that we are able to simultaneously capture up to a certain number of faults of different types. Note that the overall fault-tolerant controller operates concurrently with the plant and takes actions based on the activity in the plant and the possible faults in the controller. Information about transitions in the plant is updated by the checker which is in charge of verifying that the internal state of the redundant controller is consistent.

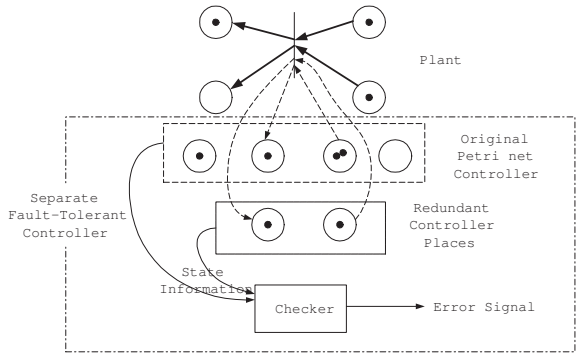

Fig. 1. Separate fault-tolerant Petri net controller.

Note that we can also slightly modify the conceptual design of the separate redundant controller in Fig.1 and allow the checker to provide the enable/disable signal for transitions. In this case, the connections from places to transitions are only used to update the number of tokens in the controller places (see Fig.2).

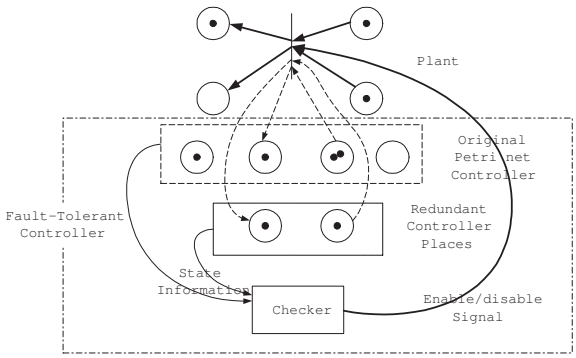

Fig. 2. Another design of separate fault-tolerant Petri net controller.

This paper is organized as follows. In Section II we review the design in [3] of Petri net controllers based on place invariants. In Section III we discuss the fault models we deal with in this paper. In Section IV we propose schemes for fault detection and identification in separate redundant Petri net controllers. We conclude and present future directions in Section V.

II. PETRI NET CONTROLLER DESIGN

Petri net controllers can appear in a variety of forms. Here we review the Petri net controllers obtained in [3]. Assuming that the plant to be controlled is modeled by a Petri net with n places and m transitions, the authors in [3] design controllers which enforce constraints of the form

$$L \cdot q \leq b, \tag{1}$$

where q is the marking vector of the Petri net, L is an $n_c \times n$ integer matrix, b is an $n_c \times 1$ integer vector, and n_c is the number of constraints. To achieve the above objective, the approach in [3] adds a controller with n_c places that are connected to and from the transitions in the original Petri net plant. The Petri net plant together with the Petri net controller form a larger Petri net with $n+n_c$ places, m transitions, marking $\begin{bmatrix} q[t] \\ q_c[t] \end{bmatrix}$, and incidence matrices $\begin{bmatrix} B^+ \\ B_c^+ \end{bmatrix}$, $\begin{bmatrix} B^- \\ B_c^- \end{bmatrix}$ and $\begin{bmatrix} B \\ B_c \end{bmatrix}$, where $B_c = B_c^+ - B_c^-$. Thinking of the n_c-dimensional marking of the controller as nonnegative slack variables, the authors of [3] write (1) as

$$L \cdot q + q_c = b, \tag{2}$$

where q_c is an $n_c \times 1$ vector which represents the marking of the controller places. Note that (2) can be used to define n_c place invariants of the form

$$\begin{bmatrix} L & I_{n_c} \end{bmatrix} \cdot \begin{bmatrix} B \\ B_c \end{bmatrix} = 0 \Leftrightarrow L \cdot B + B_c = 0 \Leftrightarrow B_c = -L \cdot B, \tag{3}$$

where I_{n_c} is the $n_c \times n_c$ identity matrix, B is the incident matrix of the plant, and B_c determines the arc weights associated with the places of the slack variables (i.e., the controller places). Note that the initial marking q_{c0} of the Petri net controller is obtained as

$$L \cdot q_0 + q_{c0} = b \Leftrightarrow q_{c0} = b - L \cdot q_0, \tag{4}$$

where q_0 is the initial marking of the plant. Note that constraints other than place invariants can also be transformed into the form of (1) (more general types of constraints are discussed in [4], [5]).

Example:

Consider a Petri net with three places $\{p_1, p_2, p_3\}$ and four transitions $\{t_1, t_2, t_3, t_4\}$ with the input/output incident matrices B^-/B^+, incident matrix B and initial marking q_0:

$$B^- = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}, \; B^+ = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 0 \\ 0 & -1 & 1 & -1 \end{bmatrix}, \; q_0 = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}.$$

If the constraints we would like to enforce (see Eq. (1)) are captured by $L = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$, the incident matrix of the corresponding controller is given by $B_c = -L \cdot B = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 0 \end{bmatrix}$. If $b = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$, then the initial marking of the controller is $q_{c0} = b - L \cdot q_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

The controller together with the original Petri net system is shown in Fig.3. Dotted lines denote arcs associated with controller places in order to be distinct from arcs in the original Petri net system.

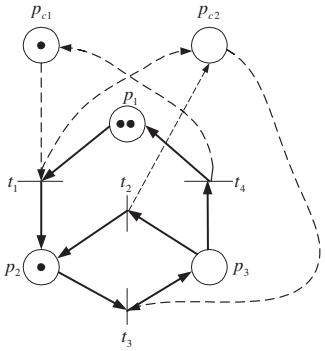

Fig. 3. The controlled Petri net of the example in Section II.

Note that the controller obtained in [3] is maximally permissive [6], i.e., the controller only acts to inhibit

transitions whose firing would cause the direct violation of the constraints in (1).

III. FAULT MODELS

In this section, we discuss the fault models that will be used in our fault detection and identification schemes. Clearly, the effectiveness of a fault model depends significantly on the particular application and the actual implementation (which varies considerably depending on the application); here we consider two different fault models.

A. Place Fault

A *place fault* models a fault that corrupts the number of tokens in a single place of the Petri net. A place fault at time epoch t results in an erroneous state $q_f[t]$ that can be expressed as

$$q_f[t] = q[t] + e_p, \tag{5}$$

where $q[t]$ is the state that would have been reached under fault-free conditions and $e_p \in Z^n$ is the place error vector. If the ith entry of e_p is negative, then the number of tokens in the ith place has decreased due to the fault; if it is positive, then the number of tokens in the ith place has increased.

B. Transition Fault

A *transition fault* models a fault in the mechanism that implements a certain Petri net transition. We say that transition t_j has a *post-condition* fault if no tokens are deposited at its output places (even though the tokens from its input places are consumed). Similarly, we say that transition t_j has a *pre-condition* fault if the tokens that are supposed to be removed from the input places are not removed (even though tokens are deposited at the corresponding output places).

Let $e_T^+ \in (Z^+)^m$ be an indicator vector of post-condition faults and $e_T^- \in (Z^+)^m$ be an indicator vector of pre-condition faults. In other words, the jth entry of e_T^+ (e_T^-) is a nonnegative integer indicating the number of post-condition (pre-condition) faults that have affected transition t_j . The erroneous state $q_f[t]$ at time epoch t can then be expressed as

$$q_f[t] = q[t] - B^+ e_T^+ + B^- e_T^-, \tag{6}$$

where $q[t]$ is the state that would have been reached under fault-free conditions, B^+ is the output incident matrix and B^- is the input incident matrix [8].

IV. FAULT DETECTION AND IDENTIFICATION USING SEPARATE REDUNDANT PETRI NET CONTROLLERS

A. Separate Redundant Petri Net Controllers

A Petri net controller such as the one used in [3] can be protected using a separate redundant Petri net controller. In a separate redundant Petri net controller, extra places and tokens are added to the original Petri net controller (see Fig.1) and their state is changed according to the transition activity in the Petri net plant. The added redundancy captures exactly the information that is necessary in order to concurrently detect and identify faults by performing linear parity checks on the combined marking of the original Petri net controller and the additional places.

More specifically, detection and identification of faults in the controlled Petri net system is achieved by adding d places in the original Petri net controller so that we obtain a separate redundant Petri net controller with $\eta \equiv n_c + d$ ($d > 0$) places and m transitions. The state evolution of this separate redundant Petri net controller is given by

$$q_h[t+1] = q_h[t] + \underbrace{\begin{bmatrix} B_c^+ \\ X^+ \end{bmatrix}}_{\mathscr{B}_c^+} x[t] - \underbrace{\begin{bmatrix} B_c^- \\ X^- \end{bmatrix}}_{\mathscr{B}_c^-} x[t], \tag{7}$$

where $x[t]$ is the firing vector (that indicates which of the m transitions in the Petri net plant fires), B_c^+ is the output incident matrix and B_c^- is the input incident matrix of the original Petri net controller. The initial number of tokens in the d redundant places and the arc weights connecting the added places with transitions in the Petri net plant (captured by X^+ and X^-) are chosen so that, at all time epochs t, $q_h[t] = \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} q_c[t]$ with C being $d \times n_c$ integer matrix to be designed.

If we combine the state evolution of the redundant and original Petri net controller, we see that

$$\begin{bmatrix} I_{n_c} \\ C \end{bmatrix} q_c[t+1] = \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} q_c[t] + \begin{bmatrix} B_c^+ \\ X^+ \end{bmatrix} x[t] - \begin{bmatrix} B_c^- \\ X^- \end{bmatrix} x[t]$$

and, since any transition t_j can potentially be enabled if the initial condition $q_c[0]$ satisfies $q_c[0] \geq B_c^-(:, j)$ (where $B_c^-(:, j)$ denotes the jth column of B_c^-), we conclude that $X^+ - X^- = C \cdot B_c^+ - C \cdot B_c^-$. Without loss of generality, we can set $X^+ = C \cdot B_c^+ - D$ and $X^- = C \cdot B_c^- - D$ for some matrix D with integer entries to be designed.

The following theorem shows that, by appropriately choosing the arc weights associated with the additional places, we can have a maximally permissive redundant Petri net controller whose state is linearly encoded in a way that enables the systematic detection and identification of faults.

Theorem 1: Under fault-free conditions, the separate redundant Petri net controller defined above has encoded state $q_h[t] = \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} q_c[t]$ for all time epochs t and is maximally permissive (i.e., it only inhibits transitions that are also inhibited by the original controller) if and only if C is a matrix with nonnegative entries and D is a $d \times m$ matrix with nonnegative entries such that $D \leq \min(CB_c^+, CB_c^-)$ (inequality is taken element-wise).

Proof:

($\Rightarrow$) The state $q_h[0] = \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} q_c[0]$ needs to have nonnegative integer entries for all valid $q_c[0]$ (a valid initial state is some vector $q_c[0]$ with nonnegative integer entries). Clearly, a necessary and sufficient condition is that C is a matrix with nonnegative integer entries.

In order for the redundant Petri net controller to admit all transition firing sequences that are allowed in the original Petri net controller, we need matrix D to have nonnegative integer entries. The proof follows easily by contradiction: Suppose D has a negative entry in its jth column and $q_c[0] = B_c^-(:,j)$; transition t_j can be fired in original Petri net controller but cannot be fired in the redundant one because $Cq_c[0] = CB_c^-(:,j) < CB_c^-(:,j) - D(:,j) = X^-(:,j)$. The requirement that $D \leq \min(CB_c^+, CB_c^-)$ follows from X^+ and X^- being matrices with nonnegative entries.

($\Leftarrow$) This direction follows easily. The only challenge is to show that if D is chosen to have nonnegative entries, then all transitions that are enabled in the original Petri net controller are also enabled in the redundant one at the same time epoch t.

If D has nonnegative entries then

$$\begin{aligned} q_c[t] \geq B_c^-(:,j) \quad &\Rightarrow \quad \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} q_c[t] \geq \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} B_c^-(:,j) \\ &\Rightarrow \quad q_h[t] \geq \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} B_c^-(:,j) \\ &\Rightarrow \quad q_h[t] \geq \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} B_c^-(:,j) - \begin{bmatrix} 0 \\ D(:,j) \end{bmatrix} \\ &\Rightarrow \quad q_h[t] \geq \mathscr{B}_c^-(:,j) \end{aligned}$$

Therefore, all transitions that are enabled in the original Petri net controller are also enabled in the redundant Petri net controller. □□

Note that the separate redundant Petri net controllers described in Theorem 1 are *bisimulation equivalent* to the original Petri net controller: any transition enabled by the original controller is also enabled by the redundant controller *and* any transition disabled by the original controller is also disabled by the redundant controller. In other words, the separate redundant controller retains the evolution properties of the original controller.

B. Fault Detection and Identification

By using the redundant Petri net controller in Theorem1, we essentially encode the original controller state $q_c[t]$ into a codeword $q_h[t]$ that consists of the original controller state and the state of the added places. The validity of a possibly invalid marking $q_f[t]$ can be checked by using the parity check matrix

$$P = \begin{bmatrix} -C & I_d \end{bmatrix} \tag{8}$$

to verify whether the *fault syndrome*, defined as

$$s[t] \equiv P \cdot q_f[t], \tag{9}$$

is equal to 0. (This is clearly true under fault-free conditions because $q_f[t] = q_h[t] = \begin{bmatrix} I_{n_c} \\ C \end{bmatrix} q_c[t]$.)

For place faults, the fault syndrome at time epoch t is given by

$$s_p[t] \equiv P \cdot q_f[t] = P \cdot (q_h[t] + e_P) = P \cdot e_P \tag{10}$$

and detection and identification is exclusively determined by matrix P. For instance, to be able to detect and identify a single place fault, we need to choose matrix C such that any two columns of matrix P are not rational multiples of each other.

For transition faults within the time epoch interval $[0,t]$, it is easy to see that the faulty state at time epoch t will be given by

$$q_f[t] = q_h[t] - \begin{bmatrix} B_c^+ \\ CB_c^+ - D \end{bmatrix} e_T^+ + \begin{bmatrix} B_c^- \\ CB_c^- - D \end{bmatrix} e_T^-, \tag{11}$$

where D is the $d \times m$ nonnegative matrix in Theorem 1. The syndrome in this case can be calculated to be [8]

$$s_T[t] = D \cdot e_T, \tag{12}$$

where $e_T = e_T^+ - e_T^-$ is an indicator vector of transition faults. Clearly, the ability to detect and identify transition faults based on the syndrome $s_T[t]$ in Eq. (12) is completely determined by matrix D. For instance, one would be able to detect a single transition fault if the columns of matrix D were distinct.

In order to detect and identify mixed place and transition faults, the approach in [8] chooses p to be a prime number larger than m (the number of transitions in the redundant system) and η (the total number of places in the redundant Petri net controller). It then defines D^*, C^*, P^* as follows

$$D^* = -p \cdot D, \tag{13}$$

$$C^* = p \cdot \mathbf{1} - C, \tag{14}$$

$$P^* = \begin{bmatrix} -C^* & I \end{bmatrix}, \tag{15}$$

where $\mathbf{1}$ is a $d \times n_c$ matrix with all entries being 1, and D is given by

$$D = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 3 & \dots & m \\ 1 & 2^2 \bmod p & 3^2 \bmod p & \dots & m^2 \bmod p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2^{d-1} \bmod p & 3^{d-1} \bmod p & \dots & m^{d-1} \bmod p \end{pmatrix}. \tag{16}$$

When $\eta = p-1$, C is chosen such that $P^* \pmod p$ satisfies

$$P_d^* = \Phi^{-1} H_d \tag{17}$$

with H_d being the parity check matrix of a Reed-Solomon code, defined in $GF(p)$ as

$$H_d = \begin{pmatrix} 1 & \alpha^1 & \alpha^2 & \alpha^3 & \dots & \alpha^{p-2} \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \dots & \alpha^{2(p-2)} \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \dots & \alpha^{3(p-2)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^d & \alpha^{2d} & \alpha^{3d} & \dots & \alpha^{d(p-2)} \end{pmatrix}, \tag{18}$$

and with matrix Φ denoting the last d columns of matrix H_d. Note that α is a primitive element in $GF(p)$, i.e., an element such that $\{1, \alpha^1, \alpha^2, \dots, \alpha^{p-2}\} = \{1, 2, 3, \dots, p-1\}$. When $\eta < p-1$, C is chosen such that $P^* \pmod p$ satisfies

$$P_d^* = \Phi^{-1} \tilde{H}_d \tag{19}$$

where $\tilde{H}_d$ is formed by the first η columns of H_d as

$$\tilde{H}_d = \begin{pmatrix} 1 & \alpha^1 & \alpha^2 & \alpha^3 & \dots & \alpha^{\eta-1} \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \dots & \alpha^{2(\eta-1)} \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \dots & \alpha^{3(\eta-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^d & \alpha^{2d} & \alpha^{3d} & \dots & \alpha^{d(\eta-1)} \end{pmatrix}. \quad (20)$$

Again, matrix Φ denotes the last d columns of matrix $\tilde{H}_d$ and α is a primitive element in $GF(p)$, i.e., an element such that $\{1,\alpha^1,\alpha^2,\dots,\alpha^{\eta-1}\} = \{1,2,3,\dots,\eta\}$. For a full description of this approach, the reader should refer to [8]. The above design satisfies $C^* > 0$ and $D^* < 0$ (element-wise) and it is different from the previous choice of matrix D, which required that the entries of D were nonnegative. Note that the design in [8] guarantees the properness of the redundant Petri net embedding in the sense that the marking of the additional places is nonnegative, and the arc weights associated with the additional places (given by $C^*B_c^- - D^*$ and $C^*B_c^+ - D^*$) are nonnegative.

Since matrix D^* has negative entries, the resulting separate redundant controller is not necessarily bisimulation equivalent to the original Petri net controller. However, the additional places do not need to function as controllers and this problem can be avoided by artificially adding a sufficiently large number of tokens to each additional place (note that this extra number of tokens should be ignored when performing parity checks). Under this slight modification the controller remains bisimulation equivalent and retains its fault detection and identification capabilities.

Mixed transition and place faults lead to the following fault syndrome at time epoch t:

$$\begin{aligned} s[t] = P^* \cdot q_f[t] &= P^* \cdot (q_h[t] - \mathscr{B}_c^+ e_T^+ + \mathscr{B}_c^- e_T^- + e_P) \\ &= D^* \cdot e_T + P^* \cdot e_P. \end{aligned} \quad (21)$$

We easily see that the syndrome $s[t] = P^* \cdot q_f[t]$ at time epoch t satisfies

$$s_p \equiv s[t] \equiv \begin{bmatrix} C^* & I \end{bmatrix} \cdot e_P \pmod{p}. \quad (22)$$

Left multiplying by Φ on both sides of (22), we obtain the modified syndrome

$$s'_p = \Phi s_p \equiv \Phi \begin{bmatrix} C & I \end{bmatrix} \cdot e_P \equiv H \cdot e_P \pmod{p}, \quad (23)$$

where H is given by H_d in (18) or $\tilde{H}_d$ in (20) depending on the value of η.

Once place faults have been successfully identified and e_P is obtained, we compute

$$s_T = (s[t] - P^* \cdot e_P)/p = (D^*/p) \cdot e_T = -D \cdot e_T \quad (24)$$

and use it to identify up to $d-1$ transition faults. Actually, with d redundant places it is possible to simultaneously identify $d-1$ transition faults and $\lfloor d/2 \rfloor$ place faults [8].

C. Examples

For simplicity, in the examples below we show how to detect and identify a single place fault or a single transition fault using the methods described in this section.

1) Detection and Identification of a Single Place Fault: Here we consider the Petri net controller discussed in the example of Section II and we assume that no transition fault happens. If we choose

$$C = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix},$$

the corresponding redundant controller has incident matrix

$$B'_c = \begin{bmatrix} B_c \\ C \cdot B_c \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 0 \\ 0 & 1 & -1 & 1 \\ 1 & 2 & -2 & 1 \end{bmatrix}$$

and initial marking $q_{c0} = \begin{bmatrix} b - Lq_0 \\ C \cdot b - CLq_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$. Note that the choice of matrix C ensures that the columns of the parity check P are not rational multiples of each other (which guarantees detection and identification of single place faults).

We now describe how to detect and identify a single place fault. The incident matrix of the original controller is $B_c = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 0 \end{bmatrix}$, with $B_c^+ = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$ and $B_c^- = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$. Since the choice of matrix D is not critical in the detection and identification of place faults, we set it to be $D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$. With this choice, matrices $\mathscr{B}_c^+$ and $\mathscr{B}_c^-$ are given by

$$\mathscr{B}_c^+ = \begin{bmatrix} B_c^+ \\ C \cdot B_c^+ - D \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 2 & 0 & 1 \end{bmatrix},$$

$$\mathscr{B}_c^- = \begin{bmatrix} B_c^- \\ C \cdot B_c^- - D \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 \end{bmatrix},$$

and the parity check is performed through the matrix

$$P = \begin{bmatrix} -C & I_2 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 1 & 0 \\ -1 & -2 & 0 & 1 \end{bmatrix}.$$

If the result is a multiple of $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ (respectively $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$, $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$), then place P_{c1} (respectively P_{c2},P_{c3},P_{c4}) has suffered a place fault.

2) Detection and Identification of a Single Transition Fault: Here we assume that no place fault happens in this situation and use two additional places to detect and identify a single transition fault. Since $B_c^+ = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$ and $B_c^- = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$, transitions t_2, t_3 and t_4 do not have

input or output arcs associated with the controller places; thus, it is impossible to find a nonnegative matrix D so that it satisfy $CB_c^+ - D \geq 0$ and $CB_c^- - D \geq 0$, and allow the detection and identification of transition faults. In this situation, we can choose D^*, C^*, P^* as discussed earlier to detect and identify a single transition fault.

We need two redundant places ($d = 2$) to detect and identify a single transition fault. From Fig. 3 we can see that there are two controller places ($n_c = 2$) and four transitions ($m = 4$), so the smallest prime number that is greater than both $m = 4$ and $\eta = 2+2 = 4$ is 5. We choose D^* as in Eq. (13) so that

$$D^* = -5 \times \left[\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2^2 & 3^2 & 4^2 \end{pmatrix} \pmod 5 \right] = \begin{pmatrix} -5 & -10 & -15 & -20 \\ -5 & -20 & -20 & -5 \end{pmatrix}.$$

Note that the first row of matrix D is redundant when $k = 1$ [8], so we choose the second and third row of Eq. (16) here. Since 3 is a primitive element in $GF(5)$, and $\eta = p - 1$, matrix H can be chosen as

$$H = \begin{pmatrix} 1 & 3 & 3^2 & 3^3 \\ 1 & 3^2 & 3^4 & 3^6 \end{pmatrix} \pmod 5 = \begin{pmatrix} 1 & 3 & 4 & 2 \\ 1 & 4 & 1 & 4 \end{pmatrix} = \underbrace{\begin{pmatrix} 4 & 2 \\ 1 & 4 \end{pmatrix}}_{\Phi} \cdot \underbrace{\begin{pmatrix} 3 & 1 & 1 & 0 \\ 2 & 2 & 0 & 1 \end{pmatrix}}_{[C\ I]}.$$

According to Eq. (14), we have $C^* = 5 \times \mathbf{1} - \begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 4 \\ 3 & 3 \end{pmatrix}$; thus, the arc weights to (from) the redundant controller places from (to) the transitions in the plant are given by

$$C^* B_c^+ - D^* = \begin{pmatrix} 9 & 14 & 15 & 22 \\ 8 & 23 & 20 & 8 \end{pmatrix},$$

$$C^* B_c^- - D^* = \begin{pmatrix} 7 & 10 & 19 & 20 \\ 8 & 20 & 23 & 5 \end{pmatrix}.$$

Furthermore, the initial marking is

$$q_{c0} = \begin{bmatrix} b \\ C^* \cdot b \end{bmatrix} - L' q_0 = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 3 \end{bmatrix}.$$

Detection and identification of a single transition fault is achieved by checking the syndrome $P^* q_f[k] = \begin{bmatrix} -C^* & I_2 \end{bmatrix} \cdot q_f[k] = \begin{bmatrix} -2 & -4 & 1 & 0 \\ -3 & -3 & 0 & 1 \end{bmatrix} \cdot q_f[k]$. If the result is a multiple of a column of matrix $D^* = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2^2 & 3^2 & 4^2 \end{pmatrix} \pmod 5$, we conclude that the corresponding transition has suffered a fault. More specifically, if the syndrome is $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ (respectively $\begin{bmatrix} 2 \\ 4 \end{bmatrix}$, $\begin{bmatrix} 3 \\ 4 \end{bmatrix}$, $\begin{bmatrix} 4 \\ 1 \end{bmatrix}$), then transition t_1 (respectively t_2, t_3, t_4) has failed to perform its post-conditions; if the result is $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$ (respectively $\begin{bmatrix} -2 \\ -4 \end{bmatrix}$, $\begin{bmatrix} -3 \\ -4 \end{bmatrix}$, $\begin{bmatrix} -4 \\ -1 \end{bmatrix}$), then transition t_1 (respectively t_2, t_3, t_4) has failed to perform its pre-conditions.

V. CONCLUSIONS AND FUTURE WORKS

In this paper we developed a methodology for detection and identification of place and transition faults in Petri net controllers. Redundant places are added to the original Petri net controller in a way that encodes information and enables fault detection and identification to be performed using algebraic techniques. Our construction of separate redundant Petri net controllers guarantees that the corresponding controllers remain bisimulation equivalent to the original one, i.e., the redundant places do not inhibit any transition which would otherwise be enabled by the original controller (and vice-versa). Necessary and sufficient conditions for bisimulation equivalence of the original controller in the case of separate controller are also proposed. In our examples, fault detection and identification is achieved by properly choosing matrices C and D. Although we focused on single faults, multiple faults may be handled in a similar fashion.

An interesting question for the future is how to optimize the parameters in the nonseparate case. Another extension of our work is to develop schemes for Petri nets with uncontrollable and unobservable transitions. Finally, it would be interesting to design distributed/hierarchical monitoring schemes for large Petri net systems by enforcing specific constraints on the Petri net embeddings.

REFERENCES

[1] A. Giua, F. DiCesare and M. Silva, "Generalized mutual exclusion constraints on nets with uncontrollable transitions," *Proc. IEEE Int. Conf. Systems, Men, Cybernetics*, Chicago, IL, 1992, pp. 974–979.

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

[3] K. Yamalidou, J. O. Moody, M. D. Lemmon, and P. J. Antsaklis, "Feedback control of Petri nets based on place invariants," *Automatica*, vol. 32, pp. 15–28, January 1996.

[4] J. O. Moody, M. V. Iordache and P. J. Antsaklis, "Enforcement of event-based supervisory constraints using state-based methods," *Proc. 38th IEEE Int. Conf. Decision and Control*, pp. 1743–1748, 1999.

[5] M. V. Iordache and P. J. Antsaklis, "Synthesis of supervisors enforcing general linear constraints in Petri nets," *IEEE Transactions of Automatic Control*, vol. 48, no. 11, pp. 2036–2039, November 2003.

[6] J. O. Moody and P. J. Antsaklis, *Supervisory Control of Discrete Event Systems Using Petri Nets*, Kluwer Academic Publishers, 1998.

[7] C. N. Hadjicostis and G. C. Verghese, "Monitoring discrete event systems using Petri net embeddings," *Application and Theory of Petri Nets 1999 (Series Lecture Notes in Computer Science)*, vol. 1639, pp. 188–207, 1999.

[8] Y. Wu and C. N. Hadjicostis, "Non-concurrent fault identification in discrete event systems using encoded Petri net states," *Proc. 41th IEEE Int. Conf. Decision and Control*, vol. 4, pp. 4018–4023, Las Vegas, Nevada, 2002.