ISSN 1392–124X (print), ISSN 2335–884X (online) INFORMATION TECHNOLOGY AND CONTROL, 2015, T. 44, Nr. 3

Partial Reconfiguration of Control Systems using Petri Nets Structural Redundancy

Mildreth Alcaraz-Mejia, Raul Campos-Rodriguez

Department of Electronic, Systems and Informatics, ITESO University, 45604 Jalisco, Mexico
e-mail: {mildreth}@iteso.mx

Ernesto Lopez-Mellado, Antonio Ramirez-Trevino

CINVESTAV Guadalajara Unit, Av. Cientifica No. 1145, Col. El Bajio, 45015, Zapopan, Jalisco, Mexico

crossref http://dx.doi.org/10.5755/j01.itc.44.3.8783

Abstract. This paper deals with the partial reconfiguration of the discrete control systems due to resource failures using the structural redundancy of the global system model. The approach herein proposed introduces a new subclass of Interpreted Petri Nets (IPN), named Interpreted Machines with Resources (IMR), allowing representing both the behaviour of a system and the resource allocation. Based on this model, an efficient reconfiguration algorithm is proposed; it is based on finding the set of all redundant sequences using alternative resources. The advantages of this structural reconfiguration method are: (1) it provides minimal reconfiguration to the system control assuring the properties of the original control system, (2) since the model includes resource allocation, it can be applied to a variety of systems such as Business Processes, and FPGAs, among others, (3) it takes advantage of the implied features of Petri net models, such as structural analysis and graphical visualization of the system and control. The method is illustrated through a case study that deals with a manufacturing system controller, which includes both alternative resources and operation sequences.

Keywords: Discrete Events, Control Systems, Reconfiguration, Redundancy, Petri nets.

1. Introduction

During the design of controllers for complex discrete event processes, one must take into account that some resources may not be available temporarily due possible failures or scheduled maintenance operations. Thus the controller must assure the process operation by using alternative resources. This feature can be achieved by executing a controller reconfiguration procedure. A variety of discrete event processes may require such a capability, namely manufacturing systems, business processes, FPGAs, and embedded systems. In such systems, alternative resources and operation sequences can be found when there exists some redundancy in the controller model; then a reconfiguration of the controller can be done to keep the system in operation. Although this work focuses on reconfigurable discrete manufacturing systems, the analysed techniques can be applied to other discrete event processes.

Reconfigurable Manufacturing Systems (RMS) have been introduced by Koren et al. in [1, 2]; they are defined as adaptable systems allowing adding, removing or modifying processes, controllers, structure of machines, to rapidly respond to evolving technology besides the market demand. RMS includes reconfigurable machines which provide flexibility in material routing. The technique here introduced provides support to analyse the redundancies given by these reconfigurable machines and for sequencing and coordination control for large RMS.

Reconfiguration techniques focusing mainly on RMS have been introduced through varied perspectives. Huang and Hsiung in [3] presented a framework for verification and estimation of dynamically partially reconfigurable systems that translate UML models into timed automata suitable for model checking. Leitão et al. in [4] presented a bio-inspired multi-agent system for RMS; the authors review the state of the art related to bio-inspired applications on manufacturing engineering problems; furthermore, they justify the use of bio-inspired agents in RMS, and enhance the need of more information about the technique in order to use it. Wang and Koren in [5] presented a scalable planning methodology for RMS using an optimization algorithm based on genetic algorithms such that the goal is minimizing the economical part of the system reconfiguration.

Petri nets (PN) have been widely used first of all for modelling and analysis of manufacturing systems [6, 7, 8]. Therefore, a natural use for PN was for the designing and implementation of the control for the automation of manufacturing systems [9, 10, 11, 12, 13, 14, 15]. There are many advantages on the use of PN for RMS, some of them are due to the inherent properties of PN such as graphical visualization and the mathematical model, i.e. an intuitive model besides the strong mathematical basis.

The approach herein proposed uses a PN subclass named Interpreted Machines with Resources (IMR) to represent both, production sequences and how resources are assigned to tasks along the production sequences. Based on the structure of a PN model, this work studies functional redundancies, e.g. different ways to obtain the same product, or different tasks sequences to meet the same goal. The need to change the current executing sequence can be due mainly to the unavailability of a resource r_i. In such a case, the redundancies are used to choose a new sequence (named recovery or alternative sequence) from those included in the production sequences to produce the same products which avoid the use of resource r_i. This work presents the controller reconfigurability property and characterizes it using the information given by the redundancies and the production sequences. When the system is reconfigurable, the recovery sequence can be computed to partially modify the controller, avoiding the use of the damage resource, whilst the production goals are reached. The advantages of this structural reconfiguration technique for the control systems based on Petri nets are: (1) the reconfiguration is minimal and preserves the properties of the initial structural control system, (2) since the model comprises resources allocation, it can be applied to other systems such as Business Processing, FPGAs, Embedded Systems, among others, (3) takes advantage of the implied features of Petri net models, such as structural analysis and graphical visualization of the system and control.

The paper is organized as follows. Section 2 presents the Interpreted PN (IPN) basic concepts. Section 3 reviews the Output Regulation Control (ORC) basic notions. Section 4 introduces the proposed definition and characterization of redundancies in a PN structure. Section 5 presents the proposed definition and characterization of the reconfigurability property and the proposed reconfiguration controller algorithm. Section 6 presents an illustrative example showing the use and advantages of this proposed technique. Finally, the conclusions and future work are presented.

2. Background on Interpreted Petri nets

This section overviews the Interpreted Petri Net (IPN) basic concepts and notation used through this paper. First, the basic Petri nets notions are introduced.

2.1. Petri nets

Definition 1. *An ordinary Petri Net structure G is a bipartite digraph represented by the 4-tuple $G=(P,T,I,O)$ where: $P=\{p_1,p_2,\dots,p_n\}$ is a finite set of vertices named places, $T=\{t_1,t_2,\dots,t_m\}$ is a finite set of vertices named transitions, $I:P\times T\to Z^+\cup\{0\}$ is a function representing the arcs going from places to transitions, $O:P\times T\to Z^+\cup\{0\}$ is a function representing the arcs going from transitions to places.*

Pictorially, places are represented by circles, transitions are represented by rectangles, and arcs are depicted as arrows. The symbol ${}^\bullet x$, $x\in P\cup T$, denotes the set of all nodes y such that $I(x,y)\neq 0$ and $x^\bullet$ $,x\in P\cup T$, denotes the set of all nodes y such that $O(x,y)\neq 0$. Let $X\subseteq P\cup T$, then ${}^\bullet X$ denotes the set of all nodes y such that $I(x,y)\neq 0$ for every $x\in X$ and $X^\bullet$ denotes the set of all nodes y such that $O(x,y)\neq 0$ for every $x\in X$.

The pre-incidence matrix of G is $C^-=[c_{ij}^-]=I(p_i,t_j)$; the post-incidence matrix of G is $C^+=[c_{ij}^+]=O(p_i,t_j)$; the incidence matrix of G is $C=C^+-C^-$. The marking function $M:P\to\mathbf{Z}^+$ represents the number of tokens (depicted as dots) residing inside each place, where $\mathbf{Z}^+$ represents the set of non-negative integers.

Definition 2. *A Petri Net system or Petri Net (PN) is the pair (G,M_0), where G is a PN structure and M_0 is the initial token distribution over places.*

Example 1. Fig. 2(a) shows a Petri net structure where:

- $P=\{p_1,p_2,\dots,p_8\}$,
- $T=\{t_1,t_2,\dots,t_6\}$,

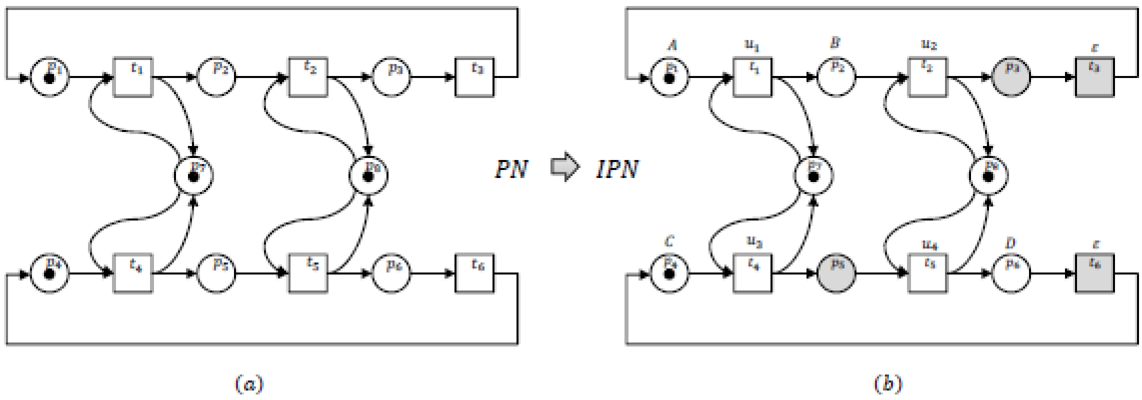


Figure 1. (a) Petri net example; (b) Interpreted Petri net example

- $$C = C^{+} - C^{-} = \begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

- The initial marking $M_0 = [1 \ \ 0 \ \ 0 \ \ 1 \ \ 0 \ \ 0 \ \ 0 \ \ 0]^T$.

2.2. Petri net structures

Definition 3. *A P-invariant Y (T-invariant X) of a PN is a rational-valued solution of equation $Y^T C = 0$ $(CX = 0)$. A P-semiflow Y (T-semiflow X) of a PN is a non-negative integer solution of the equation $Y^T C = 0$ $(CX = 0)$. A basis of minimal T-semiflows (P-semiflows) of a PN structure G is denoted $\tau(G)$ $(\rho(G))$.*

Definition 4. *The support of the vector Z representing transitions or places, denoted as $\|Z\|$, is defined as the set $\| Z \| = \{z_i \mid Z(i) \neq 0\}$.*

Definition 5. *The support of a sequence σ, denoted as $\langle\sigma\rangle$, is defined as the set $\langle\sigma\rangle = \{t_i, t_j, \ldots, t_l \mid \sigma = t_i t_j \ldots t_l\}$.*

Definition 6. *Let G be a PN structure. The induced subnet given by X, $X \subseteq P$, denoted as $[X]$ is a PN structure described by $[X] = (X, T', I', O')$ where $I' \subseteq I$ and $O' \subseteq O$ such that $I' : X \times T' \cap I$, $O' : X \times T' \cap O$ and $T' = {}^{\bullet}X \cap X^{\bullet}$. Similarly, the induced subnet given by Y, $Y \subseteq T$, denoted as $[Y]$ is a PN structure described by $[Y] = (P', Y, I', O')$ where $I' \subseteq I$ and $O' \subseteq O$ such that $I' : P' \times Y \cap I$, $O' : P' \times Y \cap O$ and $P' = {}^{\bullet}Y \cap Y^{\bullet}$.*

Definition 7. *Let $G_1 = (P_1, T_1, I_1, O_1)$ and $G_2 = (P_2, T_2, I_2, O_2)$ be two PN structures. The union of G_1 and G_2, denoted as $G_1 \cup G_2$, is performed as: $G_1 \cup G_2 = (P_1 \cup P_2, T_1 \cup T_2, I_1 \cup I_2, O_1 \cup O_2)$.*

Definition 8. *A PN system (G, M_0) is a state machine (SM) if $|{}^{\bullet}t| = 1 = |t^{\bullet}|$ for every transition t. Let G be a PN structure. A selection place $p_k \in P$ holds that $|p_k{}^{\bullet}| > 1$. An attribution place $p_l \in P$ holds that $|{}^{\bullet}p| > 1$.*

2.3. Interpreted Petri nets

An Interpreted Petri Net (IPN) [16] is a PN system including input and output information.

Definition 9. *An Interpreted Petri Net IPN is the pair (Q, M_0) such that $Q = (G, \Sigma, \lambda, \varphi)$ where:*

- G is a PN structure.
- $\Sigma = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ is the input alphabet of the net, where α_i is an input symbol.
- $\lambda : T \to \Sigma \cup \{\varepsilon\}$ is a labelling function of transitions with the following constraint:
  - $\forall t_j, t_k \in T$, $j \neq k$ if $\forall p_i$ $I(p_i, t_j) = I(p_i, t_k) \neq 0$ and both $\lambda(t_j)$, $\lambda(t_k) \neq \varepsilon$, then $\lambda(t_j) \neq \lambda(t_k)$. In this case ε represents an internal system event.

- φ is a $q \times n$ matrix, such that $y_k = \varphi M_k$ is mapping the marking M_k into the $q-dimensional$ observation vector. A column $\varphi(\bullet, i)$ is the elementary vector e_h if place p_i has associated the sensor place h, or the null vector if p_i has no associated sensor. In this case, an elementary vector e_h is the vector $q-dimensional$ with all its entries equal to zero, except entry h, that it is equal to 1. A null vector has all its entries equal to zero.

A transition $t_j \in T$ of an IPN is enabled at marking M_k if $\forall p_i \in P, M_k(p_i) \geq I(p_i, t_j)$. An enabled transition t_j, labeled with a symbol other than ε (empty or silent) symbol, must be fired when $\lambda(t_j)$ is activated. An enabled transition t_j, labeled with a ε symbol can be fired. When an enabled transition t_j is fired in a marking M_k, then a new marking M_{k+1} is reached. This fact is represented as $M_k \xrightarrow{t_j} M_{k+1}$; M_{k+1} can be computed using the dynamic part of the state equation:

$$\begin{aligned} M_{k+1} &= M_k + C \cdot v_k \\ y_k &= \varphi M_k \end{aligned} \tag{1}$$

where $v_k(j) = 1$ (since t_j was fired) and $v_k(i) = 0$, $i \neq j$; and y_k is the $k-th$ observation vector of the IPN The reachability set $R(Q, M_0)$ of an IPN is the set of all possible reachable markings from M_0 firing only enabled transitions. An IPN is safe if the maximum number of tokens residing inside each place in any reachable marking is equal to one.

According to definition of functions λ and φ, transitions and places of an IPN can be classified as follows.

A transition $t \in T$ is said to be manipulated, if $\lambda(t_j) \neq \varepsilon$, and nonmanipulated, otherwise. A place $p_i \in P$ is said to be measurable if the $i-th$ column vector of φ is not null, i.e., $\varphi(\cdot, i) \neq \vec{0}$; otherwise, p_i is nonmeasurable.

Example 2. Fig. 2 (b) shows an IPN with:

- PN structure G and initial marking M_0 as in Example 1;
- $\Sigma = \{u_1, u_2, u_3, u_4\}$ assigned to t_1, t_2, t_4, t_5 by λ function, respectively, otherwise ε is assigned;
- $\varphi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ & & & & & & & \end{bmatrix}$ represented by symbols A, B, C, D.
- By λ function, t_1, t_2, t_4, t_5 are manipulable transitions; and p_1, p_2, p_4, p_6 are measurable places, by φ.
- In the net, t_1 and t_2 are both enabled at M_0. When the input symbol u_1 is given or activated in the system, then t_1 must be fired. When u_3 is given, then t_4 must be fired.

2.4. PN and IPN properties

Definition 11. *Given a $N = (G, M_0)$, and its reachability set $R(G, M_0)$, a place $p \in P$ is $B-bounded$ if $\forall M \in R(G, M_0), M(p) \leq B$, where B is a positive integer. A PN is $B-bounded$ if each place in P is $B-bounded$. If $B = 1$, the PN is said to be safe. G is structurally bounded if G is bounded given any finite initial marking M_0 [17].*

Definition 12. *A transition t is live if at any marking $M \in R(G, M_0)$, there is a sequence of transitions whose firing reaches a marking that enables t. A PN is live if every transition in it is live. A PN is structurally live if there is a finite initial marking that makes the net live [17].*

Definition 13. *A firing transition sequence of an IPN (Q, M_0) is a transition sequence $\sigma = t_i t_j \ldots t_k \ldots$ such that $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} \ldots M_w \xrightarrow{t_k} \ldots$. The set $\mathcal{L}(Q, M_0)$ of all firing transition sequences is called the firing language of (Q, M_0) defined as $\mathcal{L}(Q, M_0) = \{\sigma \mid \sigma = t_i t_j \ldots t_k \ldots \wedge M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} \ldots M_w \xrightarrow{t_k} \ldots\}$.*

Definition 14. *Let $\mathcal{L}(Q, M_0)$ be the language generated by (Q, M_0). Then $\mathcal{L}_{mid}(Q, M_0) = \{\omega \mid \exists v, z$ such that $v\omega z \in \mathcal{L}(Q, M_0)$, v, z may be empty strings$\}$.*

Definition 15. *Let (Q, M_0) be an IPN and $K \subseteq \mathcal{L}(Q, M_0)$ the language of the specification. The language K is controllable with respect to a $\mathcal{L}(Q, M_0)$ if*

$\forall t_k \in T_{NM}$, *(i.e., $\lambda(t_k) = \varepsilon$) holds that* $\overline{K}t_k \cap \mathsf{L}(Q, M_0) \subseteq \overline{K}$.

Definition 16. *Let $\sigma = t_i t_j t_k ...$ be a firing transition sequence. The Parikh vector $\vec{\sigma}: T \to (\mathbf{Z}^+)^m$ of maps every $t \in T$ to the number of occurrences of t in σ.*

3. Output Regulation Control Background

3.1. Output regulation control

The controller reconfiguration herein used for fault recovery is based on the output regulation control (ORC) approach for fully observable system states presented in [18, 19]. The ORC scheme is shown in Fig. 2. In this approach, the system is modelled by an IPN whose output is forced to track the output language (the sequence of φM_k output symbols) of other IPN modelling the specification, named *reference*. The input control u_k given to the system is computed by the controller H taking into account the marking of both, the reference and the system model. The objective of the ORC is to keep the output error (the difference between the system and reference outputs) e_k equal to zero.

Definition 17. *A system model (Q, M_0) is an IPN represented by the state equation (1). A specification or reference model $(\breve{Q}, \breve{M}_0)$ is a live and bounded IPN, whose structure is a SM in which all transitions are manipulable and all places are measurable. The state equation of a reference model is:*

$$\breve{Q} = \begin{cases} \breve{M}_{i+1} = \breve{M}_i + \breve{C} \cdot \vec{z}_i \\ \breve{y}_i = \breve{\varphi}(\breve{M}_i) \end{cases} \qquad (2)$$

where $\breve{C}$ is the incidence matrix of $\breve{Q}$; $\breve{\varphi}$ is the output function of $\breve{Q}$.

Definition 18. *Let (Q, M_0) be the IPN model of the system to be controlled. Let $(\breve{Q}, \breve{M}_0)$ be the IPN model of the specification. The ORC problem for fully observable system states consists in finding out a partial function (controller) $H: R(Q, M_0) \times R(\breve{Q}, \breve{M}_0) \times \breve{T} \to \mathsf{L}_{mid}(Q, M_0)$ where $H(\Pi(\breve{M}_{i-1}), \breve{M}_i, \breve{t}_k) = \omega_k$ such that ω_k is controllable in $(Q, \Pi(\breve{M}_i))$, $e_k = \varphi(\breve{M}_i) - \breve{\varphi}(\breve{M}_i) = 0$, $\Pi(\breve{M}_{i-1}) \xrightarrow{\omega_k} \Pi(\breve{M}_i)$, and $\breve{M}_{i-1} \xrightarrow{\breve{t}_k} \breve{M}_i$.*

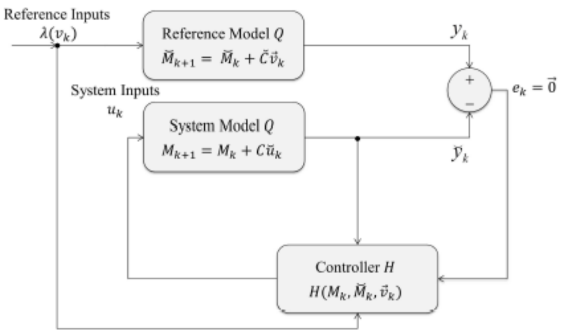


Figure 2. The ORC Architecture

The following theorem presented in [18] characterizes when the ORC problem has a solution considering the previous definitions.

Theorem 1. *Let (Q, M_0) and $(\breve{Q}, \breve{M}_0)$ be two $IPNs$ represented by Equations (1), (2), respectively. Suppose that there exists a linear function $\Pi: R(\breve{Q}, \breve{M}_0) \to R(Q, M_0)$, such that:*

1. $\Pi \breve{M}_0 = M_0$;
2. $\forall \breve{t}_m \in \breve{T}$, $\exists \omega_m \in \mathsf{L}_{mid}(Q, \Pi(\breve{M}_j))$ where $\Pi \cdot \breve{C} \cdot \vec{\breve{t}}_m = C \cdot \vec{\omega}_m$ and $\{\omega_m\}$ is controllable with respect to $(Q, \Pi(\breve{M}_j))$, with $\{\omega_m\} \subseteq \mathsf{L}(Q, \Pi(\breve{M}_j))$;
3. $\breve{\varphi}_R = \varphi_R \cdot \Pi$.

Then, the ORC problem has one solution.

Notice that the ORC is a supervisory like controller, where the specification and the system are described at different abstraction levels. Function Π translates the specification states into system states (making both models comparable with each other). The second condition of Theorem 3.1 states the controllability of system sequences. Finally, the third condition establishes that the outputs generated by both, the system and the reference, must be equal.

3.2. Solving the ORC problem

The ORC problem can be solved using the following linear programming problem (LPP) derived from Theorem 1. The problem is reduced to find out the function Π and the Parikh vectors $\vec{\omega}_m$ in order to obtain the controller H.

Algorithm 1: *Compute Π and ω.*

Input: (Q, M_0), $(\breve{Q}, \breve{M}_0)$.

Output: Π and ω matrices.

$$\begin{cases} \min & \sum_{i,j}\Pi_{ij}+\sum_{m=1}^{|\breve{T}|}\sum_{n=1}^{|T|}\vec{\omega}_m(n) \\ s.a. & \\ (C1) & \Pi\breve{M}_0=M_0 \\ (C2) & \forall\breve{t}_m\in\breve{T},\Pi\cdot\breve{C}\vec{\breve{t}}_m=C\vec{\omega}_m \\ (C3) & \varphi_R\cdot\Pi=\breve{\varphi}_R \end{cases}.$$

Notice that every $k-th$ column in Π matrix represents the marking M_k in (Q,M_0), which is related with the marking $\breve{M}_k$ in $(\breve{Q},\breve{M}_0)$ by Π function. In the same way, every $i-th$ column in ω matrix represents the Parikh vector $\vec{\omega}_i$ for the sequence ω_i in (Q,M_0), which is associated to the execution of $\breve{t}_i$ in $(\breve{Q},\breve{M}_0)$ by ω.

3.3. Compute the controller H

In order to obtain the controller H based on Π and ω, which are the outputs of the LPP in Algorithm 1, Section 3.2, use the following algorithm.

Algorithm 2: *Compute the Controller H.*

Input: (Q,M_0), $(\breve{Q},\breve{M}_0)$.

Output: Π and ω matrices.

1. For every $\breve{t}_i$ in $\breve{T}$, there exist one sequence ω_i given by Parikh vector $\breve{\omega}_i$, where $\omega_i=t_a,t_b,\ldots,t_x$. Moreover, there exists markings M_k, M_k+1, $\breve{M}_k$, $\breve{M}_{k+1}$, such that $\breve{M}_k\xrightarrow{\breve{t}_i}\breve{M}_{k+1}$ and $M_k\xrightarrow{\omega_i}M_{k+1}$, i.e. $M_k\xrightarrow{t_a}M_{k'}\xrightarrow{t_b\cdots}M_{k''}\xrightarrow{t_x}M_{k+1}$.

2. Then, compute H for every $\breve{t}_i$ in $\breve{T}$ as follows:

(a) Let $M_k\xrightarrow{\omega_i}M_{k+1}$, where $\omega_i=t_a,t_b,\ldots,t_x$. Then,

$$H(M_k,\breve{M}_{k+1},\lambda(\breve{t}_i))=\lambda(t_a)$$
$$H(M_{k'},\breve{M}_{k+1},\lambda(\breve{t}_i))=\lambda(t_b)$$
$$H(M_{k''},\breve{M}_{k+1},\lambda(\breve{t}_i))=\lambda(t_x)$$
$$H(M_{k+1},\breve{M}_{k+1},\lambda(\breve{t}_i))=\varepsilon.$$

4. Redundancies in System Models

4.1 The system modelling

An IPN model which considers the resources in the system is presented in the following example.

Example 3. Consider 5 types of machines. The first type of machine, denoted asZ_1, is able to perform sawing, drilling and routing of the raw material only in one site. Therefore, there is no need to move the material between different stations. The second type of machine, named Z_2, is a saw-drill double-function machine, which is able to cut and drill the raw material in the same site. The third type, denoted as Z_3, is an auto-feed flat-panel cutting machine, which is able to cut out raw material in different sizes. Other type of machine, named Z_4, is a one-ranged drilling machine. Finally, the last type, denoted as Z_5, is a pneumatic spindle rise router.

Fig. 3 depicts a layout of the system. The overall production line is arranged as two symmetric sections, which are Section 1 and Section 2. Section 1 is composed by three lines named Line 1, Line 2 and Line 3. Line 1 is composed by one multi-function machine of type Z_1 called $M1$. Line 2 is composed by two machines, one of type Z_2 called $M2$, and one of type Z_5 called $M3$. Some conveyors are placed between machines in order to move the material from one machine to another. Finally, Line 3 is formed by three machines, one of type Z_3 called $M4$, one of type Z_4 called $M5$, and one of type Z_5 called $M6$. As in Line 2, these machines are connected by means of two conveyors.

Moreover, the three lines are interconnected by directional conveyors that are represented as black arrows with the selection symbol $\otimes$. This set of conveyors allows to selectively change the flow of the raw material among the lines, besides, it is the mechanism used by the controller to perform control actions on the plant.

As can be seen from description of the capabilities of the different machines, the three lines are able to perform the same job over the incoming raw material. For example, Line 3, which is composed by machines $M10$, $M11$, and $M12$ of type Z_3, Z_4, and Z_5, respectively, is able to perform the cutting, the drilling and the routing of raw material. These operations can also be performed by the multiple-function machine $M1$ in Line 1, which is of type Z_1. Additionally, Line 2 is able to perform the same three operations with the combination of $M1$ and $M2$.

The system includes a set of three vertical conveyors interconnecting equivalent lines in the different sections. This allows the movement of material from Section 1 to Section 2 and the opposite. The overall system layout gives a great flexibility in the functionality of the whole system, e.g., in case of a failure of one machine, this one can be replaced by at least one different machine, in order to continue with the same production plan.

The layout is complemented by two final conveyors that collect the finished parts from the lines and put them into the inventory of final product. As mentioned before, Section 2 is a mirror of Section 1.

One simple methodology to model systems with resources is to divide the modelling in two stages: 1) The process sequences and 2) The available resources.

Stage 1. Each task τ_k, as part of the production sequence s_r, is represented by a PN that is formed by two transitions t_i^k, t_j^k, and one place $p_k^{s_r}$; transition $t_i^k(t_j^k)$ represents the start (ending) of task τ_k. Two arcs, ($t_i^k, p_k^{s_r}$) and ($p_k^{s_r}, t_j^k$), must be added to the PN. In order to obtain the model of the production sequence s_r, the final transition t_j^k of task τ_k must be merged with the initial transition t_i^{k+1} of task τ_{k+1}; where τ_{k+1} immediately follows the task τ_k in production sequence s_r. The global model of the production sequence S_{gm} is obtained by merging all places $p_k^{s_r}$ that represent the same task τ_k, from all the different production sequences. *Stage 2.* All the resources r_t (machines, robots, conveyors, etc.) represented by places p_t and arcs (p_t, t_i^k),(t_j^k, p_t) should be joined to the S_{gm}, if task τ_k is performed by resource r_t. The result is the global process plan model P_{pm}.

Fig. 4 depicts a Petri Net model that represents the production system. The place p_{43} represents the availability of raw material in the inventory, and is also the start point of the production process. Notice that p_{43} does not represent the amount of raw material but only that there exists raw material to be processed. The place p_{44} represents the final product inventory, and is the end of the production process. Again, p_{44} does not represent the amount of final products but only that a final product has been finished. The transition t_{45} that connects places p_{44} and p_{43} has no physical meaning. Nevertheless, it is fired when the system has produced a final product, in order to restart the production process.

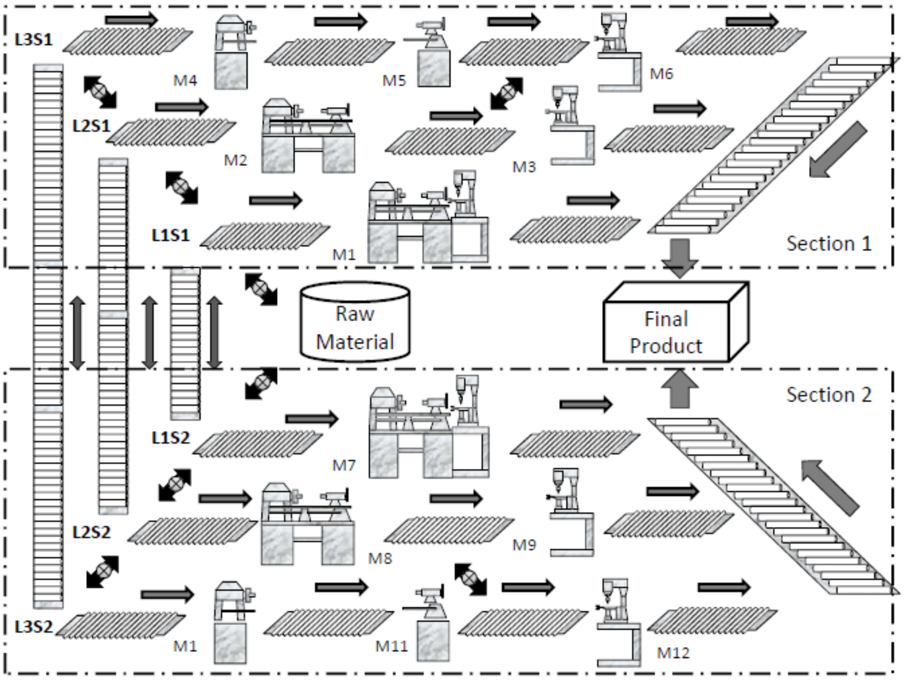


Figure 3. System Layout

In Section 1, Line 1 is formed by places p_1, p_2, p_3, p_{16} and transitions t_1, t_2, t_{35}. The place p_{16} represents the availability of machine $M1$, and place p_2 represents that $M1$ is performing the tasks over the raw material. Line 2 is formed by places p_4, p_5, p_6, p_7, p_8 and transitions t_3, t_4, t_5, t_6. The machines $M2$ and $M3$, available in this line, are represented by places p_{17} and p_{18}, respectively. The Line 3 is formed by places $p_9, p_{10}, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}$ and transitions $t_7, t_8, t_9, t_{10}, t_{11}, t_{12}$. The machines $M4$, $M5$ and $M6$, available in the line, are represented by places p_{19}, p_{20} and p_{21}, respectively. The transitions $t_{13}, t_{14}, t_{15}, t_{16}, t_{17}$ represent the interconnection of the lines in Section 1 by directional conveyors. Finally, transitions $t_{39}, t_{40}, t_{41}, t_{42}, t_{43}, t_{44}$, represent the conveyors that interconnect the lines in Section 1 with their equivalent lines in Section 2. The subnet that represents Section 2, is symmetrically arranged to Section 1, as shown in the figure.

In the net, there exist non-manipulated transitions which are guided by the internal dynamic of the system. For example, all the transitions that represent the end of the tasks performed by the machines are non-manipulated. This makes sense since the end of these tasks depend on the dynamics of each machine, which may vary over the time. On the other hand, all the places are considered measurable, which in this case, means that each stage of the production system includes a sensor.

The incidence matrix, initial marking and output function that represent the system model are depicted in Fig. 4.1.

The requirement for the plant is simple, and is represented by the net of Fig. 7. This net is interpreted as follows: when a token is moved from place $p1$ to$p2$, by the firing of transition$t1$, then it means that a final product must be produced by the system. The firing of $t2$ represents that the system is ready for the next operation.

4.2. Petri nets with resource places

The model presented above is a special class ofPN, where the S_{gm} is a state machine and the P_{pm} introduces some extra places. The resulting PN class is named *State Machines with Resource places* (SMR). Next definition formalizes the SMR class of nets.

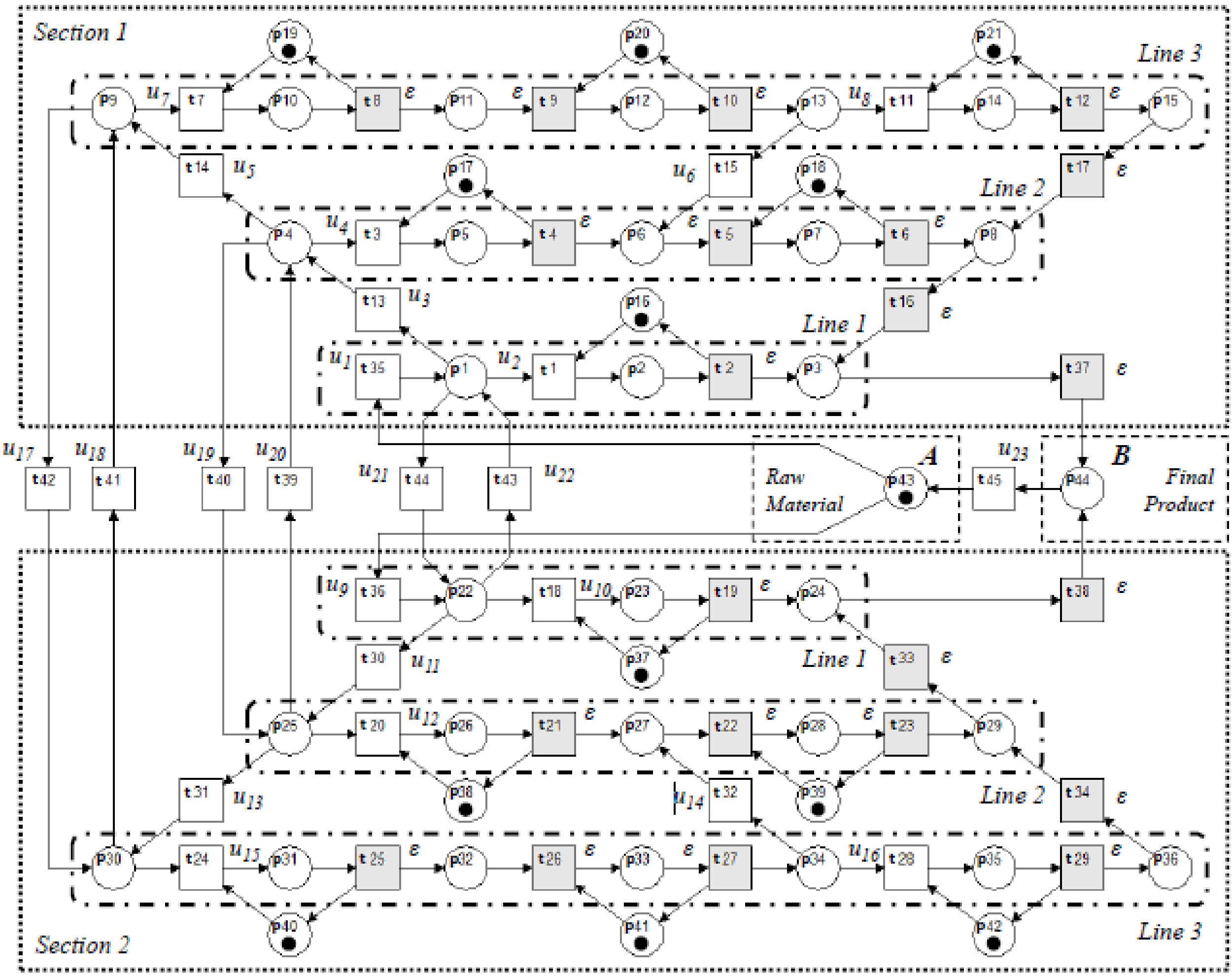


Figure 4. Petri Net Model of the Manufacturing System

Definition 19*. A State Machine with Resource Places* (SMR) *is a PN system* (G, M_0) *where:*

1. $P = P^R \cup P^{NR}$, and $P^R \cap P^{NR} = \varnothing$, where P^R is the set of places representing resources.

2. $\left[P^{NR}\right]$ is a family of $P-$components which are live and safe (SM).

3. Every $p_r \in P^R$ holds that:

[leftmargin=1.2cm]

a) $^\bullet(^\bullet p_r) \cap P = (p_r^\bullet)^\bullet \cap P \neq \varnothing$, i.e., every input place to the input transitions of p_r is also an output place to the output transitions from p_r

b) $^\bullet p_r \cap p_r{}^\bullet = \varnothing$, i.e., input transitions for $^\bullet p_r$ are not output transitions of $p_r^\bullet$.

c) $M_0(p_r) > 0$.

4. $\forall p_j$ where $\left|p_j^\bullet\right| > 1$, if $t_j \in p_j^\bullet$, then $\lambda(t_j) \neq \varepsilon$, i.e., all transitions that are outputs of selection places must be manipulables.

Figure 5. System Model: Incidence Matrix, Initial Marking and Output Symbols

Notice that the SMR (G,M_0) has the same T-invariants as its underlying SM. An PN (Q,M_0), whose structure G is a SMR, is named *Interpreted State Machine with Resource places* (IMR).

4.3. Redundancies

The flexibility given by resource redundancy, can be exploited to cope with failures in its components, downtime for maintenance, or just to change the process sequence. Informally, two sequences are redundant with each other, in terms of a Petri net, if they evolve from the same initial marking to the same final marking, and during their evolution they do not mark the same places. A formal definition is given below.

Definition 20. *Let (G,M_0) be a live and safe SM. Let σ_x, $\sigma_y \in L_{mid}(Q,M_0)$ be two fireable sequences in the PN. Let $[X]=(P_x,X,I_x,O_x)$ and $[Y]=(P_y,Y,I_y,O_y)$ be the induced subnets given by $X=\langle\sigma_x\rangle$ and $Y=\langle\sigma_y\rangle$, i.e. induced by the Parikh vectors of sequences σ_x and σ_y. Let $M_i, M_j \in R(G,M_0)$ be two reachable markings in the net. The transition sequence σ_x is redundant to σ_y and σ_y is redundant to σ_x from M_i to M_j, if $M_i \xrightarrow{\sigma_x} M_j$ and $M_i \xrightarrow{\sigma_y} M_j$ and $X \cap Y = \varnothing$ and $P_x \cap P_y = \varnothing$.*

When a transition sequence σ_x is redundant to σ_y from a marking M_i to a marking M_j the difference of their Parikh vectors $\vec{\sigma}_x - \vec{\sigma}_y$ is a T-invariant resulting from linear combination of semipositive T-invariants. This fact is stated below.

Proposition 1. Let (N,M_0) be a live PN. If σ_x is redundant to σ_y from M_i to M_j, then $\vec{\sigma}_x - \vec{\sigma}_y$ is a T-invariant.

Proof. Since σ_x is redundant to σ_y from M_i to M_j, it holds that $M_i \xrightarrow{\sigma_x} M_j$ and $M_i \xrightarrow{\sigma_y} M_j$ then $M_i + C\cdot\vec{\sigma}_x = M_i + C\cdot\vec{\sigma}_y$. Thus $C\cdot\vec{\sigma}_x = C\cdot\vec{\sigma}_y$, this is, $C \cdot (\vec{\sigma}_x - \vec{\sigma}_y) = 0$. Therefore $(\vec{\sigma}_x - \vec{\sigma}_y)$ is a T-invariant.

However, not all T-invariants are formed from redundant sequences. Then, in general, they cannot be computed from the PN structure. Fortunately, if the PN is an SMR (or IMR model), then redundancies can be computed from the PN structure, leading to polynomial algorithms to compute such redundancies. Below this observation is formalized.

Definition 21. *Let (G, M_0) be a live and safe SM. Let $\tau(G)$ be a basis of T-semiflows of the SM. Let $\tau_i, \tau_j \in \tau(G)$. Let $[\|\tau_i\|] = (P_i, \|\tau_i\|, I_i, O_i)$ and $[\|\tau_j\|] = (P_j, \|\tau_j\|, I_j, O_j)$ be the induced subnets from T-semiflows τ_i, τ_j, respectively. The set of redundancy vectors is $Rds(G) = \{Rds_k \mid Rds_k = \tau_i - \tau_j$, for all $j > i$, such that $P_i \cap P_j$ includes just one selection place p_k and one attribution place p_l in $[\|\tau_i\|] \cup [\|\tau_j\|]\}$.*

The algebraic T-semiflow basis in a SM can be determined using d different T-covertures, where d is the dimension of the T-invariant basis. Now, the following algorithm provides one way to find out the set of redundancy vectors.

Algorithm 3: *Computation of set $Rds(G)$*

Inputs: $\tau(G)$, a basis of minimal T-semiflows.
Outputs: $Rds(G)$, set of redundancy vectors.

1. Let $Rds(G) = \%$.
2. Compute the $t-$components for every pair τ_i, τ_2 as follows (see Definition 2.2): $T_i = \|\tau_i\|$, $P_i = {}^{\bullet}T_i \cap T_i^{\bullet}$, $I_i = P_i \times T_i \cap I$, $O_i = P_i \times T_i \cap O$ and $T_j = \|\tau_j\|$, $P_j = {}^{\bullet}T_j \cap T_j^{\bullet}$, $I_j = P_j \times T_j \cap I$, $O_j = P_j \times T_j \cap O$.
3. Compute $P_{\cap ij} = P_i \cap P_j$ and $P_{\cup ij} = P_i \cup P_j$.
4. If $P_{\cap ij} \cap X = \{p_k\}$ and $P_{\cap ij} \cap Y = \{p_l\}$, then $Rds(G) \leftarrow Rds(G) \cup \{\tau_i - \tau_j\}$ where:
   a) $X = \{p_h \mid p_h{}^{\bullet} \cap (T_i \cup T_j) > 1$ and $p_h \in P_{\cup ij}\}$
   b) $Y = \{p_h \mid {}^{\bullet}p_h \cap (T_i \cup T_j) > 1$ and $p_h \in P_{\cup ij}\}$

Notice that previous algorithm has polynomial computational complexity. Now, from $Rds(G)$ all the redundancies in the IRM model are obtained. Let us first introduce the following notation. X^+ and X^- denote the positive and negative entries of the vector X, respectively, as follows:

$$X^+[i] = \begin{cases} 1, & \text{if} X[i] = 1 \\ 0, & \text{otherwise} \end{cases}$$

$$X^-[i] = \begin{cases} 1, & \text{if} X[i] = -1 \\ 0, & \text{otherwise} \end{cases}$$

The next proposition exploits the information from the vectors X^+ and X^- of $X \in Rds(G)$ for obtaining the redundancies of the IMR.

Proposition 2. Let (G, M_0) be a live and safe SM. Let $X \in Rds(G)$ such that $\vec{\sigma}_x = X^+$; $\vec{\sigma}_y = X^-$. Then there exist fireable redundant sequences σ_x, σ_y.

Proof. Since $\vec{\sigma}_x - \vec{\sigma}_y \in Rds(G)$ and $Rds(G)$ is generated by some linear combinations (positive and negative) of T-semiflows, then $\vec{\sigma}_x - \vec{\sigma}_y$ is a T-invariant; i.e.

$$C(\vec{\sigma}_x - \vec{\sigma}_y) = 0. \tag{3}$$

Moreover $\vec{\sigma}_x \square \tau_i$ and $\vec{\sigma}_y \square \tau_j$, where τ_i, τ_j are T-semiflows. Since the SM is live and bounded, the T-semiflows τ_i, τ_j are obtained from fireable sequences α_i and α_j, respectively. Thus, the projections of α_i and α_j over the transitions included in $\vec{\sigma}_x$ and $\vec{\sigma}_y$ lead to the fireable sequences σ_x and σ_y.

Thus, from equation (3) it is obtained $M_i - M_i + C(\vec{\sigma}_x - \vec{\sigma}_y) = 0$, or $M_i + C\vec{\sigma}_x = M_i + C\vec{\sigma}_y = M_j$. Then $M_i \xrightarrow{\sigma_x} M_j$ and $M_i \xrightarrow{\sigma_y} M_j$.

Since the vectors in $Rds(G)$ obtained from the difference of two T-semiflows where the common transitions to both T-semiflows are eliminated, and in $SM$the transitions have only one output or input place, then $[\|\vec{\sigma}_x\|]$ and $[\|\vec{\sigma}_y\|]$ do not have common transitions nor places. Thus, they meet the redundancy definition.

Proposition 2 leads to the following algorithm to compute the fireable sequences σ_x and σ_y from X^+ and X^- of a redundancy vector $X \in Rds(G)$

Algorithm 4: *Compute the fireable sequences σ_x and σ_y from $X \in Rds(G)$*

Inputs: (G, M_0) with $G = (P, T, I, O)$; X^+, X^-.
Outputs: σ_x, σ_y

1. Build the induced subnets for X^+ and X^- as follows (see Definition 2.2): $T_x = \|X^+\|$, $P_x = {}^{\bullet}T_x \cap T_x^{\bullet}$, $I_x = P_x \times T_x \cap I$, $O_x = P_x \times T_x \cap O$ and $T_y = \|X^-\|$, $P_y = {}^{\bullet}T_y \cap T_y^{\bullet}$, $I_y = P_y \times T_y \cap I$, $O_y = P_y \times T_y \cap O$.
2. Construct the sequences σ_x, σ_y using the structures given by (P_x, T_x, I_x, O_x) and (T_y, P_y, I_y, O_y)

Notice that the complexity of previous algorithm is polynomial. Thus, the computation of the redundancies can be performed in polynomial time.

5. Reconfigurable Controllers

This section presents an extension to the ORC scheme to include fault recovery capabilities. It introduces the concept of controller reconfiguration and its characterization. In addition, it presents a procedure to perform the controller reconfiguration in a faulty scenario, based on the original controller H.

5.1. Reconfigurable ORC Scheme

In order to properly cope with the fault recovery problem, two modules are added to the ORC scheme shown in Fig. 2. The ORC with Reconfiguration scheme is showed in Fig. 6. When a fault occurs in the system, the Diagnoser D detects the error. Then, D sends the error information included in the faulty vector K (defined below) to the Reconfigurer E, which indicates the places representing faulty resources.

Definition 22. *The faulty resource vector K of a system model (Q, M_0) is a vector of size $|P|$ such that:*

$$K[i] = \begin{cases} 1, & \text{if } p_i \text{ is a faulty place} \\ 0, & \text{otherwise} \end{cases}$$

$\forall i \in [1, |P|]$, where a faulty place represents a resource diagnosed in fault by D.

Definition 23. *The faulty transitions vector F of a system model (Q, M_0) is a vector of size $|T|$ such that:*

$$F[i] = \begin{cases} 1, & \text{if } t_i \in {}^\bullet p_j \cup p_j{}^\bullet \text{ where } p_j \text{ is a faulty place} \\ 0, & \text{otherwise} \end{cases}$$

$\forall i \in [1, |T|]$, where a transition t_i such that $F[i] = 1$ is called a faulty transition.

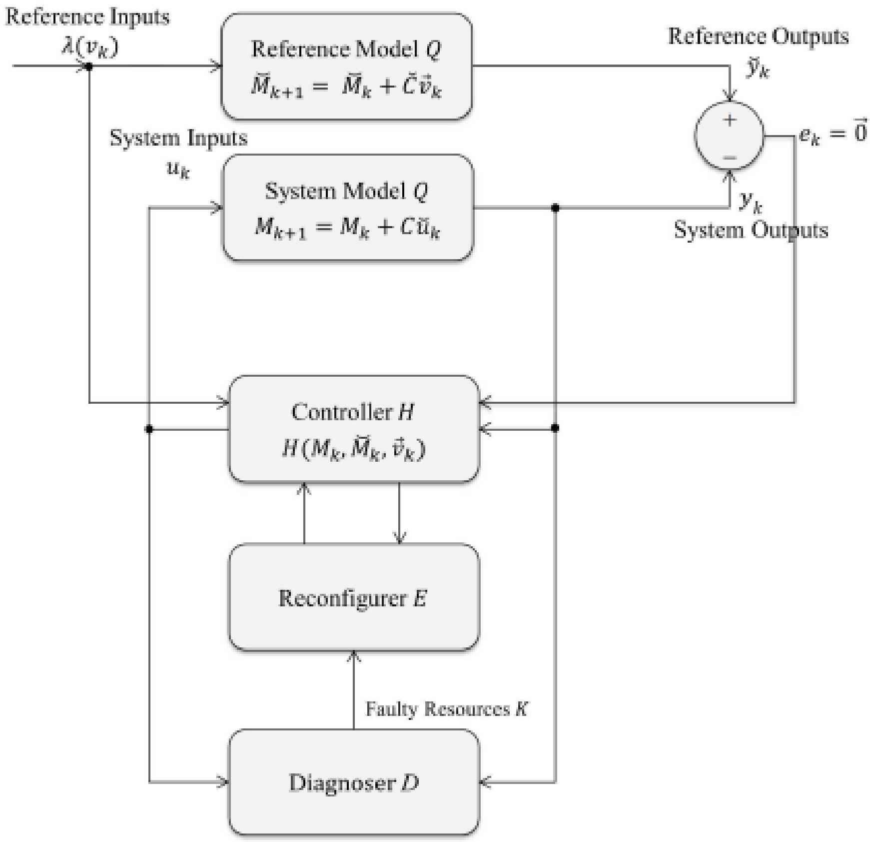


Figure 6. The ORC scheme with reconfiguration

5.2. Reconfiguration of the Controller

This section describes the controller reconfiguration technique, which is based on system redundancies. The reconfigurability property is defined and characterized, and then a procedure for partial reconfiguration is derived.

Definition 24. *Let (Q, M_0) be a live IMR system model of fault-free behaviour. Let $(\breve{Q}, \breve{M}_0)$ be a reference model. Let H be the controller solution for the ORC defined by Q and $\breve{Q}$. Let F be the* *faulty transitions vector and $\|F\|$ its support. The controller H is said to be reconfigurable with respect to F if $\forall t_f \in \|F\|$ and for all sequence σ including t_f, where $\alpha\sigma\gamma \in Im(H)$, there exist a controllable sequence $\sigma' \in L_{mid}(Q, M_0)$ redundant to σ such that $\langle\sigma'\rangle \cap \|F\| = \varnothing$.*

In the following, the characterization of the fault recovery problem for an ORC scheme with reconfiguration of the controller is presented.

***Theorem 2**. Let (Q, M_0) be a live IMR system model. Let $(\breve{Q}, \breve{M}_0)$ be a reference model. Let H be the controller given for the function Π and the Parikh vectors $\vec{\omega}_m$, solution of the ORC problem for Q and $\breve{Q}$. Let F be the faulty transitions vector and $\|F\|$ its support.*

If the controller H is Reconfigurable with respect to F then the fault recovery problem has a solution.

Proof. Let $H(\Pi\breve{M}_j, \breve{M}_i, \breve{t}_k) = \omega_k$ such that $\langle\omega_k\rangle \cap \|F\| \neq \varnothing$, then there exists $\omega_k' = \alpha\beta'\gamma$ such that $\omega_k = \alpha\beta\gamma$ and $\langle\omega_k'\rangle \cap \|F\| = \varnothing$, where β' is redundant to β. Therefore, $C(\vec{\beta} - \vec{\beta}') = 0$ by Proposition 4.3. Thus $C(\vec{\omega}_k - \vec{\omega}_k') = 0$, and then $C \cdot \vec{\omega}_k = C \cdot \vec{\omega}_k'$ because $(\vec{\omega}_k - \vec{\omega}_k') = (\vec{\beta} - \vec{\beta}')$. Since $\breve{M}_j \xrightarrow{\breve{t}_k} \breve{M}_i$ and $\Pi\breve{M}_j \xrightarrow{\omega_k} \Pi\breve{M}_i$, then $\breve{M}_i = \breve{M}_j + \breve{C} \cdot \vec{\breve{t}}_k$ and $\Pi\breve{M}_i = \Pi\breve{M}_j + C \cdot \vec{\omega}_k$. By Theorem 3.1, $\breve{M}_i = \breve{M}_j + \breve{C} \cdot \vec{\breve{t}}_k$ is equivalent under the function Π to $\Pi\breve{M}_i = \Pi\breve{M}_j + \Pi\breve{C} \cdot \vec{\breve{t}}_k$. As $C \cdot \vec{\omega}_k = C \cdot \vec{\omega}_k'$. then $\Pi\breve{M}_i = \Pi\breve{M}_j + C \cdot \vec{\omega}_k$ is equivalent to $\Pi\breve{M}_i = \Pi\breve{M}_j + C \cdot \vec{\omega}_k'$. Therefore, $\Pi\breve{C} \cdot \vec{\breve{t}}_k = C \cdot \vec{\omega}_k'$. Furthermore, $\vec{\omega}_k'$ is controllable since $\vec{\omega}_k$ was controllable and β' is controllable. Thus, Condition 2 of the Theorem 3.1 is satisfied. Since Conditions 1 and 3 hold as well, then the controller H' defined as:

$$H'(\Pi\breve{M}_{i-1}, \breve{M}_i, \breve{t}_k) = \begin{cases} \omega_k, & \text{If } \langle\omega_k\rangle \cap \|F\| = \varnothing; \\ \omega_k', & \text{otherwise.} \end{cases}$$

solves the ORC.

The proof of the previous theorem states that the specified behaviour by the reference $(\breve{Q}, \breve{M}_0)$ still holds. At the same time, the use of faulty resources (faulty transitions) is avoided.

5.3. Reconfiguration procedure

Based on the proof of the previous theorem the following reconfiguration algorithm for the controller can be derived.

Table 1. Function Π

	$k-th$ vector of Π
Π_1	$[00000000000000011111100000000000000011111110]^T$
Π_2	$[00000000000000011111100000000000000011111101]^T$

Table 2. Parikh vectors ω

	$i-th$ vector of ω
ω_1	$[110000000000000000000000000000000010100000000]^T$
ω_2	$[000000000000000000000000000000000000000000001]^T$

Table 3. Controller$H$

M_k	$\breve{M}_k$	$\breve{t}_k$	Controller Sequence	Fired System Sequence
$[0000000000000001111110000000000000001111110]^T$	$[10]^T$	t_1	u_1	t_{35}
$[1000000000000001111110000000000000001111100]^T$	$[10]^T$	t_1	u_2	t_1
$[0100000000000001111110000000000000001111100]^T$	$[10]^T$	t_1	ε	t_2
$[0010000000000001111110000000000000001111100]^T$	$[10]^T$	t_1	ε	t_{37}
$[0000000000000001111110000000000000001111101]^T$	$[10]^T$	t_1	ε	ε
$[0000000000000001111110000000000000001111101]^T$	$[01]^T$	t_2	u_{23}	t_{45}
$[0000000000000001111110000000000000001111110]^T$	$[01]^T$	t_2	ε	ε

Algorithm 5:*Reconfiguration Procedure;*

Inputs:

F: Faulty transitions vector,

H:: The controller (the partial function),

$Rds(G)$: The set of redundancy vectors,

(Q, M_0): System model (Q, M_0).

Outputs:

σ_y : The redundancy sequence,

$\overleftarrow{\sigma}_y$: The redundancy Parikh vector of σ_y,

H' : The reconfigured controller

1. $\forall R \in Rds(G)$ such that $F^T \bullet R \neq 0$ do

   a) If $F^T \bullet R^+ \neq 0$ then using Algorithm 4:

      i. Compute the sequence σ_x with R^+.

      ii. Compute the sequence σ_y, with R^-.

   else

      i. Compute the sequence σ_y, with R^+.

      ii. Compute the sequence σ_x with R^-.

2. $\forall w_m = r\sigma_x s$ such that $w_m = H(M_k, \breve{M}_k, t_i)$, redefine $w_m' = H'(M_k, \breve{M}_k, t_i)$, where $w_m' = r\sigma_y s$.

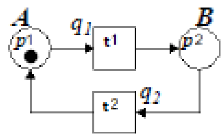
3. $Rds(G) = Rds(G) - R$.

6. Illustrative Example

Assume that the system presented in Example 3, Section 4.1, must follow the reference depicted in Fig. 1. The reference is simple, and is interpreted as follows: when a token is moved from place p_1 to p_2, by the firing of transitiont_1, then it means that a final product must be produced by the system. The firing of t_2 represents that the system is ready for the next operation. The incidence matrix, the initial marking and the output function of the reference model are depicted in Fig. 1, along with the model.

Applying Algorithm 1 to the given system model and reference model shown in Fig. 4 and 7, respectively, the LPP provides Π and ω for the solution to the ORC problem as stated in Theorem 1. Π and Γ are shown in Table 1 and 2. The controller can now be computed using the Algorithm 2. The resulting controller H is shown in Table 3.

The matrices Π and ω found by the LPP are not the unique solution to the ORC. The set of redundancy vectors, computed with Algorithm 3, can be used to construct any other solution to the problem. Since the redundancy vectors are closely related with the null-space of the incidence matrix, there exist an infinite number of them. Fortunately, there is no need to compute all these vectors at once, since a basis of minimal T-semiflows of G includes all the information about the redundancies in the system. In fact, under the case of a fault in the system, a linear combination of the vectors in that basis can be used to compute a required redundancy vector.



$$\breve{C} = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}, \quad \breve{M}0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \varphi = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Figure 7. The Reference Model

Table 4. A basis of minimal T-semiflows $\tau(G)$

τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_{10}	τ_{11}	τ_{12}	τ_{13}	τ_{14}
0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
1	1	0	0	0	0	0	0	1	1	0	0	0	0
1	1	0	0	0	0	0	0	1	1	0	0	0	0
1	0	1	0	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0
1	1	1	0	0	0	0	0	1	1	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	1	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	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	1	1	0	0	0	0
0	0	0	1	0	1	0	0	0	1	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	1	1
0	0	0	1	1	1	1	0	1	1	0	0	0	0
1	1	1	0	0	0	0	0	1	1	0	0	0	1
0	0	0	1	1	1	1	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	1	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	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	0
1	1	1	1	1	1	1	0	1	1	0	0	1	1

A basis of minimal T-semiflows is shown in Table 4, where every vector τ_i, $1 \leq i \leq 14$, represents a redundancy as stated in Proposition 1. Notice that the t-semiflow represented by column 14, say τ_{14} describes the flow from marking M_0 through the firing of transitions $t_{35}, t_1, t_2, t_{37}, t_{45}$ until the same marking M_0. Also, the t-semiflow represented by column 2, say τ_2, describes a flow from M_0 to M_0 but now through the firing of transitions $t_{35}, t_{13}, t_3, t_4, t_5, t_6, t_{16}, t_{37}, t_{45}$ which represents a different task path in the system. Observe that the induced subnets given by the vectors from column 2 and column 14, share one selection place p_1 and one attribution place p_3. Then, as dictated by Algorithm 3, $\tau_{14} - \tau_2$ represents a redundancy vector:

$$Rds_1(G) = \tau_{14} - \tau_2 = \begin{cases} 1, & \text{for} i = 1,2. \\ -1, & \text{for} i = 3-6, 13, 16. \\ 0, & \text{otherwise.} \end{cases}$$

Now, assume that the faulty vector (see Definition 5.1 in Section 5) is:

$$F[l] == \begin{cases} 1, \text{ for} i = 1,2. \\ 0, \text{otherwise.} \end{cases}$$

In other words, the faulty transitions are t_1 and t_2. Thus, the support of vector F is $\|F\| = \{t_1, t_2\}$.

Then, applying the Algorithm 4 in Section 4 for the faulty vector F with the information given by Rds_1, the faulty sequence $\sigma_x = t_2 t_3$ and the recovering sequence $\sigma_y = t_4 t_5 t_6$ are obtained. Therefore, the new reconfigured controller H' is presented in Table 5, which avoids the use of faulty transitions using an alternative sequence (route) ω_1'.

Table 5. Reconfigured Controller H'

M_k	$\check{M}_k$	$\check{t}_k$	Controller Sequence	Fired System Sequence
$[00000000000000011111100000000000000011111110]^T$	$[10]^T$	t_1	u_1	t_{35}
$[10000000000000011111100000000000000011111100]^T$	$[10]^T$	t_1	u_3	t_{13}
$[00001000000000010111100000000000000011111100]^T$	$[10]^T$	t_1	u_4	t_3
$[00000100000000011111100000000000000011111100]^T$	$[10]^T$	t_1	ε	t_5
$[00000010000000011011100000000000000011111100]^T$	$[10]^T$	t_1	ε	t_6
$[00000001000000011111100000000000000011111100]^T$	$[10]^T$	t_1	ε	t_{16}
$[00100000000000011111100000000000000011111100]^T$	$[10]^T$	t_1	ε	t_{37}
$[00000000000000011111100000000000000011111101]^T$	$[10]^T$	t_1	ε	ε
$[00000000000000011111100000000000000011111101]^T$	$[01]^T$	t_2	u_{23}	t_{45}
$[00000000000000011111100000000000000011111110]^T$	$[01]^T$	t_2	ε	ε

7. Conclusions

The paper proposed a PN approach for dealing with automated fault recovery of reconfigurable manufacturing systems. The output regulation control scheme has been extended by including controller reconfiguration capabilities. The proposed technique for reconfiguration profits of structural redundancies in the system model for determining, when there exist, alternative production sequences after a resource failure is diagnosed. Based on the redundancies, the controller is partially recomputed; then the reconfigured controller avoids the use of the faulty resource. The reconfiguration process is accomplished by polynomial algorithms, allowing on-line fault recovery; consequently such a technique is scalable to large systems in which several faults may be handled.

8. Acknowledgments

This work was partially supported by the Ministry of Science and Technology of Mexico (CONACYT) under grant no. 165095 and 157967. The authors would like to thank the reviewers and editors for their work on this paper.

References

[1] **Y. Koren, U. Heisel, F. Jovane,T. Moriwaki, G. Pritschow, G. Ulsoy, H. Van Brussel.** Reconfigurable manufacturing systems. *CIRP Annals-Manufacturing Technology*, 1999, Vol. 48, No. 2, 523–540.

[2] **Y. Koren, M. Shpitalni**. Design of reconfigurable manufacturing systems. *Journal of Manufacturing Systems*, 2010, Vol. 29, No. 4, 130-141.

[3] **C. Huang, P. Hsiung**. Model based verification and estimation framework for dynamically partially reconfigurable systems. *IEEE Transactions on Industrial Informatics*, 2011, Vol. 7, No. 2, 287-301.

[4] **P. Leitão, J. Barbosa, D. Trentesaux.** Bio-inspired multi-agent systems for reconfigurable manufacturing systems. *Engineering Applications of Artificial Intelligence*, 2012, Vol. 25, No. 5, 934-944.

[5] **W. Wang, Y. Koren.** Scalability planning for reconfigurable manufacturing systems. *Journal of Manufacturing Systems*, 2012, Vol. 31, No. 2, 83-91.

[6] **M. Silva, R. Valette.** Petri nets and flexible manufacturing. In: *Advances in Petri nets, Springer*, 1990, pp. 374-417.

[7] **N. Viswanadham, Y. Narahari.** Performance modeling of automated manufacturing systems. *Prentice Hall, Englewood Cliffs, NJ*, 1992.

[8] **F. DiCesare, G. Harhalakis, J. M. Proth, M. Silva, F. B. Vernadat.** Practice of Petri nets in manufacturing. *Springer*, 1993.

[9] **M. C. Zhou, F. DiCesare, D. L. Rudolph.** Design and implementation of a Petri net based supervisor for a flexible manufacturing system. *Automatica,* 1992, Vol. 28, No. 6, 1199-1208.

[10] **M. C. Zhou, F. DiCesare, A. Desrochers.** A hybrid methodology for synthesis of Petri net models for manufacturing systems. *IEEE Transactions on Robotics and Automation,* 1992, Vol. 8, No. 3, 350–361.

[11] **M. C. Zhou, K. Venkatesh.** Modeling, Simulation, and control of flexible manufacturing systems. Vol. 6, *World Scientific,* 1999.

[12] **M. C. Zhou, F. DiCesare.** Petri net synthesis for discrete event control of manufacturing systems. Vol. 4. *Springer Science & Business Media*, 2012.

[13] **R. David, H. Alla**. Petri nets for modeling of dynamic systems: A survey. *Automatica,* 1994, Vol. 30, No. 2, 175-202.

[14] **D. Y. Lee, F. DiCesare.** Scheduling flexible manufacturing systems using Petri nets and heuristic search. *IEEE Transactions on Robotics and Automation,* 1994, Vol. 10, No. 2, 123-132.

[15] **K. Yamalidou, J. Moody, M. Lemmon, P. Antsaklis.** Feedback control of Petri nets based on place invariants. *Automatica,* 1996, Vol. 32, No. 1, 15-28.

[16] **A. Ramirez-Treviño, I. Rivera-Rangel, E. Lopez-Mellado.** Observability of discrete event systems modeled by interpreted Petri nets. *IEEE Transactions on Robotics and Automation,* 2003, Vol. 19, No. 4, 557-565.

[17] **J. Desel, J. Esparza.** Free Choice Petri Nets. *Cambridge University Press, New York, NY,* 1995.

[18] **J. F. Sanchez-Blanco, A. Ramirez-Treviño, A. Santoyo.** Regulation control in interpreted Petri nets using trace equivalence. In: *Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics*, 2004, 2, pp. 1843-1848.

[19] **R. Campos-Rodriguez, A. Ramirez-Treviño, E. Lopez-Mellado.** Regulation control of partially observed discrete event systems. In *Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics,* 10-13 Oct. 2004, Vol. 2, pp. 1837–1842.

Received November 2014.