Proceeding of the 2004 American Control Conference
Boston, Massachusetts June 30 - July 2, 2004



# Resilience to Failures and Reconfigurations in the Supervision Based on Place Invariants

Marian V. Iordache and Panos J. Antsaklis

***Abstract*— The supervision based on place invariants (SBPI) is a very efficient technique for the enforcement of linear marking constraints on Petri nets. In this paper we first outline the SBPI and the extension of the SBPI for liveness enforcement. Then we discuss the qualities and limitations of these methods from a fault tolerance/reconfigurations perspective.**

## I. Introduction

The supervision based on place invariants (SBPI) [1], [2], [3] constrains the marking $\mu$ of a plant Petri net (PN) to satisfy specifications of the form

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

where $L \in \mathbb{Z}^{n_c \times m}$, $b \in \mathbb{Z}^{n_c}$, $\mathbb{Z}$ is the set of integers, $m$ is the number of places of the PN, and $n_c$ the number of constraints. This class of specifications is powerful enough for many applications [3]. In particular, in the case of *safe* PNs any state specification can be written in the form (1) [4], [1] and the derivation of (1) from a Boolean expression can be carried out rather easily [4], [2]. Further, the problem of enforcing languages of PNs with distinct labels can be reduced to the enforcement of (1) on an enhanced plant PN [5]. This class of languages as well as the languages corresponding to (1) are neither a subset nor a superset of the regular languages.

The enforcement of constraints (1) was considered first in [1]. In [6], the computation of the supremal controllable subpredicate was applied to (1) for partially controllable PNs. Computationally efficient but suboptimal methods for the design of supervisors enforcing (1) have appeared in [3], [7], dealing not only with partial controllability but also with partial observability. There are also other approaches for various subclasses of (1), such as in [8], [9], and for marked graphs [10]. Unlike to the supervisory control developments in the Ramadge and Wonham setting [11], the issue of nonblocking supervision has not been considered in the original papers proposing the SBPI. Instead, the issues of deadlock prevention and liveness enforcement have been dealt with separately in [12], [13]. Combining specifications (1) with liveness enforcement has also been considered in [14] for resource allocation systems. Software tools implementing SBPI methods are available [15].

This paper aims to emphasize some remarkable qualities of the SBPI in the context of faults and reconfiguarations. We will discuss both the SBPI and also its extension to liveness enforcement [13], [16]. This discussion applies also to other extensions of the SBPI not considered here, such as the enforcement of extended constraints [5] and decentralized control [17]. We will show that the SBPI designs can be robust under various circumstances when the faults/reconfigurations are modeled as changes in marking by token loss/gain or changes in the parameter "$b$" of (1). We also show that in our setting the changes of controllability/observability that are not critical to a SBPI design can be easily identified.

The paper is organized as follows. In section II we introduce the SBPI. Then, the extension of the SBPI for liveness enforcement is outlined in section III. The fault/reconfiguration properties are studied in section IV.

The authors are with the Department of Electrical Engineering, University of Notre Dame, IN 46556, USA. E-mail: iordache.1, antsaklis.1@nd.edu.

The authors gratefully acknowledge the partial support of the Lockheed Martin Corporation, of the National Science Foundation (NSF ECS99-12458), and of DARPA/IXO-NEST Program (AF-F30602-01-2-0526).

## II. Outline of the SBPI

The system to be controlled is called **plant**, and is assumed to be given in the form of a PN $\mathcal{N} = (P, T, F, W)$, where $P$ is the set of places, $T$ the set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ the set of transition arcs, $W : F \rightarrow \mathbb{N}^*$ the weight function, and $\mathbb{N}^*$ the set of positive integers. The SBPI provides a supervisor enforcing (1) in the form of a PN $\mathcal{N}_s = (P_s, T, F_s, W_s)$ with

$$D_s = -LD \tag{2}$$
$$\mu_{0,s} = b - L\mu_0 \tag{3}$$

where $D_s$ is the incidence matrix of the supervisor, $\mu_{0,s}$ the initial marking of the supervisor, and $\mu_0$ the initial marking of $\mathcal{N}$. The places of the supervisor are called **control places**. The supervised system, that is the **closed-loop** system, is a PN of incidence matrix $D_c = [D^T, (-LD)^T]^T$. Note that the control places participate in the place invariants:

$$\mu_s = b - L\mu \tag{4}$$

**Example 2.1** The PN of Fig. 1(a) adapts a PN model of [3] of an unreliable machine [18], [3]. Assuming we desire to enforce

$$\mu_1 + \mu_2 + \mu_5 \leq 1 \tag{5}$$
$$\mu_3 + \mu_7 \leq 1 \tag{6}$$

by (2–3) we obtain supervisor shown in Fig. 1(b), consisting of the control places $C_1$ and $C_2$. Note that for all reachable markings we have the following place invariants

$$\mu_{C_1} = 1 - \mu_1 - \mu_2 - \mu_5 \tag{7}$$
$$\mu_{C_2} = 1 - \mu_3 - \mu_7 \tag{8}$$

0-7803-8335-4/04/$17.00 ©2004 AACC

This shows why the supervision is said to be "based on place invariants". □

The optimality of the supervision design is summarized in the following result from [3], [2]:

**Theorem 2.1** [3], [2] *If $L\mu_0 \leq b$ then the PN supervisor with incidence matrix $D_s = -LD$ and initial marking $\mu_{0,s} = b - L\mu_0$ enforces the constraint $L\mu \leq b$ when included in the closed-loop system $D_c = [D^T, D_s^T]^T$. Furthermore, the supervision is least restrictive.*

Let $\mu_c$ be the marking of the closed-loop, and let $\mu_c|_{\mathcal{N}}$ denote $\mu_c$ restricted to the plant $\mathcal{N}$. Let $t \in T$ be a transition. $t$ is **closed-loop enabled** if $\mu_c$ enables $t$. $t$ is **plant-enabled**, if $\mu_c|_{\mathcal{N}}$ enables $t$ in $\mathcal{N}$. The supervisor **detects** $t$ if $t$ is closed-loop enabled at some reachable marking $\mu_c$ and firing $t$ changes the marking of some control place. The supervisor **controls** $t$ if there is a reachable marking $\mu_c$ such that $t$ is plant-enabled but not closed-loop enabled.

In PNs with uncontrollable and unobservable transitions, admissibility issues arise. Indeed, a supervisor designed as in (2–3) may include control places preventing plant-enabled uncontrollable transitions to fire, and may contain control places with marking varied by firings of closed-loop enabled unobservable transitions. Such a supervisor is clearly not implementable. A supervisor is admissible, if it only controls controllable transitions, and it only detects observable transitions. The constraints $L\mu \leq b$ are **admissible** if the supervisor defined by (2–3) is admissible. When inadmissible, the constraints $L\mu \leq b$ are transformed (if possible) to an admissible form $L_a\mu \leq b_a$ such that

$$L_a\mu \leq b_a \Rightarrow L\mu \leq b \tag{9}$$

Then, the supervisor enforcing $L_a\mu \leq b_a$ is admissible, and enforces $L\mu \leq b$ as well.

**Example 2.2** Assume $t_2$ and $t_5$ uncontrollable in Fig. 1(a). Then $\mu_2 + \mu_5 \leq 1$ is not admissible, as enforcing it may attempt controlling either of $t_2$ and $t_5$. However, it can be checked that $\mu_1 + \mu_2 + \mu_5 \leq 1$ is admissible and $\mu_1 + \mu_2 + \mu_5 \leq 1 \Rightarrow \mu_2 + \mu_5 \leq 1$. □

Note that there are "structural" conditions that are sufficient for admissibility. They provide both a quick admissibility test and a principle for the design of the transformed constraints $L_a\mu \leq b_a$. They can be seen as inequalities in terms $L$ and $D$. Thus, to ensure that the supervisor (2–3) controls only the controllable transitions it is sufficient to require [3], [7]:

$$LD(\cdot, T_{uc}) \leq 0 \tag{10}$$

where $T_{uc}$ is the set of uncontrollable transitions. Further, to ensure that the supervisor (2–3) detects only the observable transitions it is sufficient to require [3], [7]:

$$LD(\cdot, T_{uo}) = 0 \tag{11}$$

where $T_{uo}$ is the set of unobservable transitions. Given (1) and an initial marking $\mu_0$, (10) and (11) are only sufficient for admissibility. However, if $L$ is fixed and $\mu_0$ and $b$ are variables, we have the following optimality property.

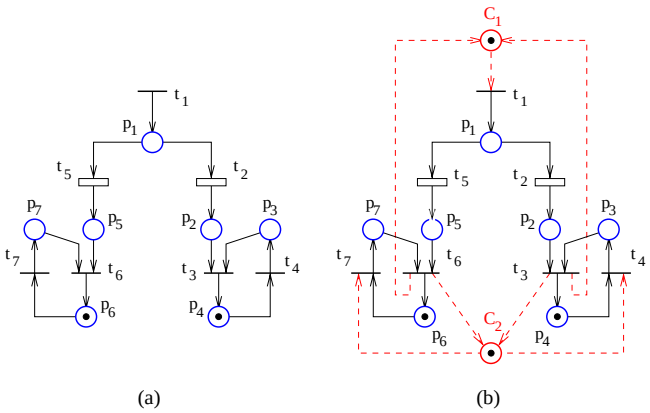



Fig. 1.

**Theorem 2.2** [16] *The supervisor of (2–3) is admissible for all $\mu_0$ and $b \geq L\mu_0$ iff $L$ satisfies (10–11).*

Note that a design robust to variable $\mu_0$ and $b$ can be of interest in the context of failures and reconfigurations. One fact that has not been noticed before is that admissibility conditions of the same type as (10–11) can be found for the more general PNs that label the PN transitions with events in a set $\Sigma$, similar to the labeling of state machines in the supervisory control setting of [11]. Such a "labeled" PN is defined as $\mathcal{N} = (P, T, F, W, \lambda)$, where $\lambda : T \to \Sigma \cup \{\varepsilon\}$ is the labeling function and $\varepsilon$ the null event. In this setting, a supervisor controls/observes events rather than transitions. Let $\Sigma_{uc}/\Sigma_{uo}$ denote the set of uncontrollable/unobservable events. (Naturally, $\varepsilon \in \Sigma_{uc}$ and $\varepsilon \in \Sigma_{uo}$.) Then the admissibility conditions can be written as:

$$\forall t_1, t_2 \in T,\ \lambda(t_1) = \lambda(t_2) \quad \Rightarrow \quad LD(\cdot, t_1) = LD(\cdot, t_2) \tag{12}$$

$$\forall t \in T,\ \lambda(t) \in \Sigma_{uc} \quad \Rightarrow \quad LD(\cdot, t) \leq 0 \tag{13}$$

$$\forall t \in T,\ \lambda(t) \in \Sigma_{uo} \quad \Rightarrow \quad LD(\cdot, t) = 0 \tag{14}$$

Note that (12–14) can be written compactly as $LA \leq 0$, for some matrix $A$. This means that methods finding $L_a$ and $b_a$ subject to (9) and (12–14) are readily available. Indeed, the methods of [3], [9] finding $L_a$ and $b_a$ such that $L_a$ satisfies (9) and (10) or (9–11) can be applied also here, by replacing (10) with $LA \leq 0$.

This indicates that the SBPI can approach very general supervisory settings. Compared to what has been considered in the PN literature [19], the setting of [20] is harder to incorporate here. It involves disabling/enabling groups of transitions, as opposed to individual transitions, while each transition being observable. Here, this would lead to nonlinear admissibility constraints.

## III. Liveness Enforcement

As shown in the previous section, several qualities of the SBPI are as follows. First, it allows the design of

supervisors enforcing (1) under very general settings. Second, the design can be carried out independently of the initial marking and $b$, the free-term of (1). This quality is of interest in the context of faults/reconfigurations, as it shows the design can be easily adapted to changes in marking and certain changes in the specifications. However, the SBPI enforcement does not ensure the supervision will avoid deadlocks. This section proposes an approach that enhances (1) with additional constraints of the same type such that liveness specifications are satisfied. Because this approach relies too on the admissibility constraints (10) and (11), the additional constraints produced are admissible, while the design is still independent of the initial marking. However, if willing to give up this quality, we could use another approach to generate the additional constraints, such as the liveness enforcement approach for fully observable and bounded PNs of [21].

Given a PN $\mathcal{N}$ of initial marking $\mu_0$, a transition $t$ is live if for all reachable markings $\mu$, there is an enabled firing sequence that includes $t$. Given $\mathcal{T} \subseteq T$, $(\mathcal{N}, \mu_0)$ is $\mathcal{T}$-live if all $t \in \mathcal{T}$ are live. Further, $(\mathcal{N}, \mu_0)$ is live if $T$-live (i.e., all transitions $t$ are live).

**Example 3.1** Note that the PN of Fig. 1(b) is not live, and not even deadlock-free: the sequence $t_1, t_2, t_7$ leads to deadlock. Here, the supervisor causes deadlock, as the plant in Fig. 1(a) is live. This section will consider enhancing a specification $L\mu \leq b$ with additional constraints $L'\mu \leq b'$ such that the resulting supervised system is live. □

A procedure for the design of $\mathcal{T}$-liveness enforcing supervisors has been proposed in [16], [13]. This is the input of the procedure:

1) A PN $\mathcal{N}$ and the set $\mathcal{T} \subseteq T$;
2) The sets of uncontrollable and unobservable transitions, $T_{uc}$ and $T_{uo}$;
3) Optionally, the set of reachable-marking constraints (RMC) $G\mu \leq h$.

Note that the RMC describe constraints that the reachable markings are known to satisfy. Formally, given a set of initial markings of interest $\mathcal{M}_I$, the RMC satisfy that $\forall \mu_0 \in \mathcal{M}_I \; \forall \mu \in \mathcal{R}(\mathcal{N}, \mu_0)$: $G\mu \leq h$, where $\mathcal{R}(\mathcal{N}, \mu_0)$ is the set of reachable markings of $(\mathcal{N}, \mu_0)$. The RMC is an optional argument, and its implicit value corresponds to $\mathbb{N}^m$ (all possible markings). The output of the procedure is the following:

1) Two sets of constraints $C\mu \leq d$ and $C_0\mu \leq d_0$, describing the supervisor.
2) A boolean variable $LR$, where $LR = TRUE$ indicates least-restrictive supervision.[1] ($LR$ is set by checking sufficient conditions for least-restrictive supervision; in principle, the supervision could be least-restrictive also when $LR = FALSE$).
3) A boolean variable $TERM$, where $TERM = TRUE$ indicates successful termination.

The role of the constraints $C\mu \leq d$ and $C_0\mu \leq d_0$ is described in the following theorem from [16], [13].

**Theorem 3.1** *If the procedure terminates and $TERM = TRUE$, then $C\mu \leq d$ is admissible and $(\mathcal{N}, \mu_0)$ supervised according to $C\mu \leq d$ is $\mathcal{T}$-live for all initial markings $\mu_0 \in \mathcal{M}_I$ satisfying $C_0\mu_0 \leq d_0$ and $C\mu_0 \leq d$.*

Note that $\mathcal{M}_I = \mathbb{N}^m$ when no RMC is given. On the other hand, when an RMC is given, the supervisor design may rely on it, and so $\mathcal{T}$-liveness enforcement is not guaranteed for $\mu_0 \notin \mathcal{M}_I$.

As Theorem 3.1 shows, the initial marking is a variable, not a given input, just as in the SBPI. In this context, this is what "least restrictive supervision" means. The supervisor defined by $C\mu \leq d$ and $C_0\mu \leq d_0$ is least restrictive if for all initial markings $\mu_0$

- if $C\mu_0 \not\leq d$ or $C_0\mu_0 \not\leq d_0$, no $\mathcal{T}$-liveness enforcing supervisor of $(\mathcal{N}_0, \mu_0)$ exists.
- if $C\mu_0 \leq d$ and $C_0\mu_0 \leq d_0$, the supervisor enforcing $C\mu \leq d$ is the least restrictive $\mathcal{T}$-liveness enforcing supervisor of $(\mathcal{N}_0, \mu_0)$.

Note that if the procedure terminates and certain sufficient conditions are satisfied, the supervisor given by $C\mu \leq d$ and $C_0\mu \leq d_0$ is guaranteed to be least restrictive. In particular, when $\mathcal{T} = T$ (full liveness enforcement), $\mathcal{N}$ is fully controllable and observable ($T_{uc} = \emptyset$ and $T_{uo} = \emptyset$) and the procedure terminates, the procedure generates the least restrictive liveness enforcement supervisor, if a liveness enforcing supervisor exists.

**Example 3.2** As shown before, enforcing the specification (5–6) on the PN of Fig. 1(a) leads to deadlock. To add new constraints that ensure liveness, we start with the PN of Fig. 2(a), corresponding to the closed-loop of Fig. 1(b). Consider applying the $\mathcal{T}$-liveness enforcing procedure with $\mathcal{T} = T$ (full liveness desired), $T_{uo} = \emptyset$ and $T_{uc} = \{t_2, t_5\}$. Due to (7–8), the RMC are $\mu_1 + \mu_2 + \mu_5 + \mu_9 = 1$ and $\mu_3 + \mu_7 + \mu_8 = 1$. The procedure terminates with the following constraints $C\mu \leq d$:

$$\mu_1 + 2\mu_2 + \mu_5 + \mu_7 + \mu_8 + \mu_9 \geq 2 \quad (15)$$

$$\mu_1 + \mu_2 + \mu_3 + 2\mu_5 + \mu_8 + \mu_9 \geq 2 \quad (16)$$

and the following constraints $C_0\mu \leq d_0$

$$\mu_3 + \mu_4 \geq 1 \quad (17)$$

$$\mu_6 + \mu_7 \geq 1 \quad (18)$$

In view of the RMC, $\mu_8$ and $\mu_9$ can be substituted, and then (15) and (16) become

$$\mu_2 - \mu_3 \geq 0 \quad (19)$$

$$\mu_5 - \mu_7 \geq 0 \quad (20)$$

The supervised PN is shown in Fig. 2(b), while Fig. 2(c) shows the original plant supervised with (5–6) and the additional constraints (19–20) for liveness enforcement. □

[1] For the simplicity of the presentation, $LR$ has not been included in the procedures of [16], [13]; however, it is implemented in the package [15].

The procedure of [16], [13] does not have guaranteed termination. In practice, the termination issue can be mitigated by using "transformations to EAC-nets" instead of "AC-nets" [16]. However, the total elimination of this issue is a matter of further research.

## IV. Fault/Reconfiguration Properties

### A. Changes in marking

Changes in marking could model failures or certain reconfigurations. For instance, in a model of a manufacturing system, loss of tokens could refer to machine breakdown, while gain of tokens to new machines being added to the system. Let $\Delta\mu$ denote the marking change by either gain or loss of tokens. In this section we deal with two questions: "When needs a marking change $\Delta\mu$ be detected?" and "How should the supervisor be updated when a marking change $\Delta\mu$ is detected?" Note the following. Given a specification $L\mu \leq b$:

1) If $L\Delta\mu \not\leq 0$, the change should be detected, or else the specification may be violated.
2) If $L\Delta\mu \leq 0$ and $L\Delta\mu \neq 0$, the only effect of an undected change may be overly restrictive supervision.
3) If $L\Delta\mu = 0$, the change of marking has no effect on the supervision.
4) If $\Delta\mu$ is detected, the marking of the control places should be updated according to $\mu_s = \mu_s - L\Delta\mu$ (see equation (4)). There are two cases
   a) if $\mu_s \not\geq 0$, the supervisor needs redesign.
   b) if $\mu_s \geq 0$, no redesign is required, as Theorem 2.1 can still be applied.

The same remarks apply also to the supervisors generated by the liveness enforcement procedure, where we have to consider $C\mu \leq d$ and $C_0\mu \leq d_0$ instead of $L\mu \leq b$ and Theorem 3.1 together with Theorem 2.1.

**Example 4.1** We consider here the manufacturing example of [12], which is shown in Fig. 3. There, to prevent deadlock, the following constraints were generated:

$$\mu_1 + \mu_6 \geq 1 \quad (21)$$
$$\mu_4 + \mu_7 \geq 1 \quad (22)$$
$$\mu_3 + \mu_4 + \mu_5 + \mu_7 + \mu_8 \geq 2 \quad (23)$$
$$2\mu_1 + \mu_3 + \mu_4 + 3\mu_5 + \mu_6 + \mu_7 + \mu_8 \geq 5 \quad (24)$$

where (21–23) correspond to $C\mu \leq d$, being implemented by $C_1 \ldots C_3$, and (24) to $C_0\mu \leq d_0$. We illustrate the remarks 1, 2, and 4 for the marking $\mu$ shown in Fig. 3:

1) If $p_1$ loses one token (a machine breaks down and becomes unusable), then $C\Delta\mu \not\leq 0$. If the change goes unnoticed, the marking of $C_1$ (which, by (21) must satisfy the invariant $\mu_{C_1} = \mu_1 + \mu_6 - 1$) is not decremented. Thus, the sequence $t_{13}t_{13}t_1t_2$ stays enabled. Firing it leads to a marking that violates (21) (creating a local deadlock too.)
2) If $p_1$ gains one token (e.g. a broken machine is fixed), then $C\Delta\mu \leq 0$ and $C\Delta\mu \neq 0$. If the change goes unnoticed, the marking of $C_1$ is not incremented. Therefore, the supervision becomes more restrictive, as the sequence $t_{13}t_{13}t_{13}t_{13}t_1t_1t_2t_2$, which is legal under the new circumstance, is not allowed by $C_1$. However, the specification remains enforced.
4) In case 1 above, the marking of $C_1$ is to be updated as $\mu_{C_1} \to \mu_{C_1} - 1$, and in case 2 as $\mu_{C_1} \to \mu_{C_1} + 1$. In both cases $\mu_{C_1} \geq 0$, and so no redesign is necessary (cf. Theorem 3.1 and Theorem 2.1). □

Changes in marking may render a specification infeasible. For instance, when a manufacturing system has lost enough many resources, it can no longer be live. In such situations attempting to redesign the supervisor is useless, as no solution exists. Two alternatives are possible: relaxing the specification, which is considered in section IV-C, and reconfiguring the system. In the PN literature, the latter approach has been used in [22] to remove the resources involved in deadlocked parts of the system and use them in other parts of the system that could continue their operation. In [22], faults are modeled by loss of tokens, as in this section.

We have assumed the initial marking to be known. In the literature, the SBPI has been adapted to the case when the initial marking is unknown in [23], based on marking estimation. This suggests also that the case in which faults are not detected could be approached in the same framework.

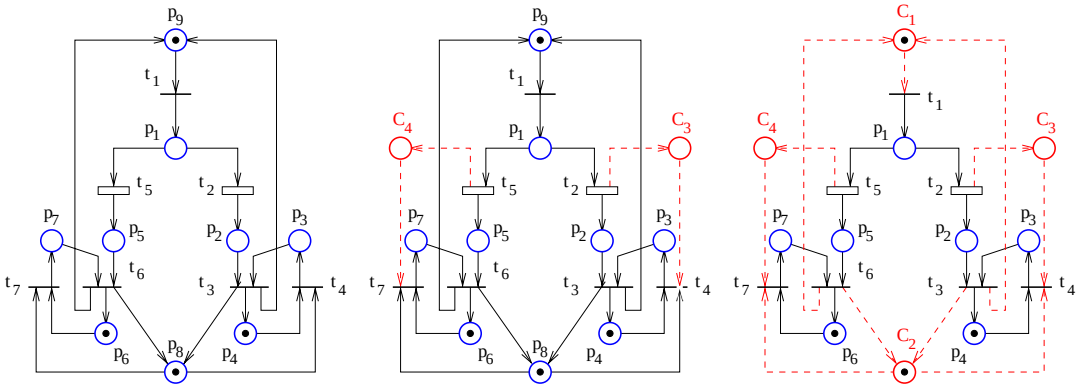
### B. Actuator/Sensor failures

We include here the failures that cause changes in the sets of uncontrollable and unobservable transitions $T_{uc}$ and $T_{uo}$. Thus, an actuator failure would increase the set of uncontrollable transitions, while a sensor failure the set of unobservable transitions. Let $T'_{uc}$ and $T'_{uo}$ be the new $T_{uc}$ and $T_{uo}$ after a failure has occurred. Two questions arising here are: "Which faults need to be detected, to ensure proper operation of the supervisor?" and "Which faults require supervisor redesign?"

The design based on the admissibility conditions (10–11) has the following qualities:

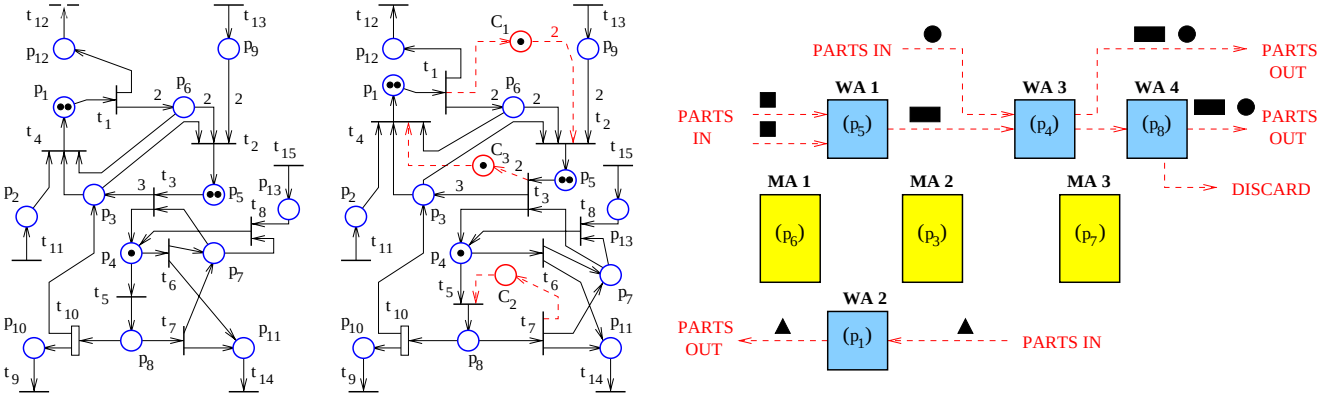
1) The admissibility conditions allow a quick (or online) identification of critical faults, where a fault is critical if it requires redesign. Indeed, given $D_s = D_s^+ - D_s^-$ the incidence matrix of the supervisor and the input and output matrices $D_s^+$ and $D_s^-$, note that:
   a) (10–11) are not affected by $t$ becoming unobservable if $D_s^+(\cdot, t) = D_s^-(\cdot, t)$.
   b) (10–11) are not affected by $t$ becoming uncontrollable if $D_s^-(\cdot, t) = 0$.

   Note that when (10–11) remain satisfied, the supervisor does not need redesign.
2) The conditions (10–11), if not satisfied after the failure, indicate also which part of the supervisor needs update.




(a) (b) (c)

Fig. 2.




(a) (b) (c)

Fig. 3.

3) The update for safety constraints $L\mu \leq b$ (but not for liveness enfocement!) may be feasible online, as there are efficient algorithms that can find admissible $L_a\mu \leq b_a$ satisfying (9) subject to (10–11); see [3], [7], [24].

In addition, note also that in our approach

4) No update is required when the observations of $t$ and $t'$ become indistinguishable if $D_s(\cdot, t) = D_s(\cdot, t')$.
5) The supervisor operates properly, whether the faults that are not critical are detected or not. However, the critical faults need to be detected.

**Example 4.2** We illustrate here the points 1, 2, and 4 on Fig. 1(b) and the constraints (5–6). Note that $C_1$ corresponds to (5) and $C_2$ to (6).

1) Since none of $t_2$ and $t_5$ is connected to either of $C_1$ and $C_2$, the fault leading to $t_2$ and $t_5$ uncontrollable or unobservable is not critical. Also, since $C_1, C_2 \notin \bullet t_6$, the fault leading to $t_6$ uncontrollable is not critical.
2) If $t_7$ becomes uncontrollable, only the implementation of (6) is affected.
4) If the observations of $t_6$ and $t_3$ become indistinguishable, the supervisor is not affected, as $t_6$ and $t_3$ have the same effect on $C_1$ and $C_2$: both $C_1$ and $C_2$ receive one token when either of $t_6$ or $t_3$ fires. □

### C. Changes in desired constraints

Changes in the specification may arise in various situations, for instance, in the context of the supervisor redesign of section IV-A. It may be that after certain faults the specification has become infeasible, and so it needs to be relaxed in order to be able to redesign the supervisor. The changes may involve both the specification $L\mu \leq b$ and the liveness specification. We do not consider here the latter, as it could hardly be handled online by our approach. For changes in the specification $L\mu \leq b$ there are two cases:

1) A specification $L\mu \leq b$ is replaced by $L'\mu \leq b'$.
2) A specification $L\mu \leq b$ is replaced by $L\mu \leq b'$ (only $b$ changes.)

There are two other possibilities:

A. liveness requirements are present

B. no liveness requirements are present

By combining the cases 1 and 2 with A and B we have four possibilities. Note that 1B may be approachable online, due to the efficient methods of [3], [7], [24]. Further, case 2B involves the following. Let $\mu_s = \mu_s + b' - b$ be the updated marking of the control places (see equation (4)). Then:

1) If $\mu_s \geq 0$, no supervisor update is necessary (Theorem 2.1).
2) If $\mu_s \ngeq 0$, the supervisor needs to be redesigned.

Case 2A can be treated online when $L\mu + \mu_s = b$ is not included in the RMC. Then, nothing changes in the liveness enforcing supervisor: we only need to check that the updated $\mu_s$ together with the plant marking satisfy $C\mu \leq d$ and $C_0\mu \leq d_0$. On the other hand, the case when $L\mu+\mu_s = b$ is included in the RMC is considerably more difficult, as the RMC may affect the design of the liveness enforcement.

### D. Incorporating failures/reconfigurations in the model

Certain failures or reconfigurations may be incorporated in a PN model. An example is the manufacturing system from [12] shown in Fig. 3. A reconfiguration/failure situation incorporated in the model is as follows. When a machine $MA3$ is idle, it corresponds to a token in $p_7$. A machine $MA3$ can be used in the work areas $WA3$ ($p_4$) and $WA4$ ($p_8$). However, a failure is possible when the machine is in $WA4$, which is modeled by the uncontrollable transition $t_{10}$. When the failure occurs, the part the machine was working on is discarded ($t_9$) and the machine reallocated ($t_{10}$) to be used in $WA1$ or $WA2$.

The SBPI can naturally approach such models by modeling failures/reconfigurations as uncontrollable (and/or unobservable) transitions. However, for deadlock prevention and liveness enforcement, one needs procedures that can deal with irreversible processes in the model. For instance, in Fig. 3, the transitions $t_5$, $t_9$, and $t_{10}$ can fire only finitely many times, regardless of the initial marking. The $\mathcal{T}$-liveness enforcement procedure we have proposed meets this need, as we can include in $\mathcal{T}$ only the part of the system we are interested in making live. For instance, we can exclude from $\mathcal{T}$ transitions modeling failures. Further, $\mathcal{T}$ can be adjusted automatically during the supervisor design process to remove transitions that cannot be made live [16].

## V. Conclusions

In this paper we have discussed the SBPI and some related approaches relying on the SBPI, with application to the context of systems with faults and reconfigurations. We have shown these approaches to have some remarkable qualities that may lead to robust designs, in which only minor updates, if any, are required in case of faults/reconfigurations. Thus we have shown that some of the faults/reconfigurations that can be handled particularly well are those modeled by (a) token loss/gain; (b) certain changes in the form of the constraints; (c) certain changes in the controllability/observability of the system. This performance is due to the following: (i) the supervisor design is independent of the initial marking of the system and also of the "$b$" parameter of the specification; (ii) the versatility of the structural admissibility conditions.

## References


[1] A. Giua, F. DiCesare, and M. Silva, "Generalized mutual exclusion constraints on nets with uncontrollable transitions," in *Proc. IEEE Internat. Conf. Syst., Man, Cybern.*, 1992, pp. 974–979.

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

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

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

[5] M. V. Iordache and P. J. Antsaklis, "Synthesis of supervisors enforcing general linear vector constraints in Petri nets," in *Proc. 2002 Amer. Contr. Conf.*, 2002, pp. 154–159.

[6] Y. Li and W. Wonham, "Control of Vector Discrete-Event Systems II - Controller Synthesis," *IEEE Trans Automat. Contr.*, vol. 39, no. 3, pp. 512–530, 1994.

[7] J. O. Moody and P. J. Antsaklis, "Petri net supervisors for DES with uncontrollable and unobservable transitions," *IEEE Trans Automat. Contr.*, vol. 45, no. 3, pp. 462–476, 2000.

[8] H. Chen, "Control synthesis of Petri nets based on s-decreases," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 10, no. 3, pp. 233–250, 2000.

[9] G. Stremersch, *Supervision of Petri Nets*. Kluwer, 2001.

[10] P. Darondeau and X. Xie, "Linear control of live marked graphs," *Automatica*, vol. 39, no. 3, pp. 429–440, 2003.

[11] P. Ramadge and W. Wonham, "The control of discrete event systems," *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, 1989.

[12] M. V. Iordache, J. O. Moody, and P. J. Antsaklis, "Synthesis of deadlock prevention supervisors using Petri nets," *IEEE Trans. Robot. Automat.*, vol. 18, no. 1, pp. 59–68, Feb. 2002.

[13] M. Iordache and P. Antsaklis, "Design of T-liveness enforcing supervisors in Petri nets," *IEEE Trans Automat. Contr.*, vol. 48, no. 11, pp. 1962–1974, 2003.

[14] J. Park and S. Reveliotis, "Liveness-enforcing supervision for resource allocation systems with uncontrollable behavior and forbidden states," *IEEE Trans. Robot. Automat.*, vol. 18, no. 2, pp. 234–240, 2002.

[15] M. V. Iordache and P. J. Antsaklis, "Software tools for the supervisory control of Petri nets based on place invariants," University of Notre Dame," Technical report isis-2002-003, Apr. 2002.

[16] M. V. Iordache, "Methods for the supervisory control of concurrent systems based on Petri net abstractions," Ph.D. dissertation, University of Notre Dame, 2003.

[17] M. V. Iordache and P. J. Antsaklis, "Decentralized control of Petri nets," in *Proceedings of the Workshop on Discrete Event Systems Control, of the International Conference on the Application and Theory of Petri Nets (ATPN 2003)*, 2003, pp. 143–158.

[18] A. Desrochers and R. Al'Jaar, *Applications of Petri nets in Manufacturing Systems: Modelling, Control and Performance Analysis*. IEEE Press, 1995.

[19] L. E. Holloway, B. H. Krogh, and A. Giua, "A survey of Petri net methods for controlled discrete event systems." *Discrete Event Dynamic Systems*, vol. 7, no. 2, pp. 151–190, 1997.

[20] L. Holloway and B. Krogh, "Synthesis of feedback control logic for a class of controlled Petri nets," *IEEE Trans Automat. Contr.*, vol. 35, no. 5, pp. 514–523, 1990.

[21] K. He and M. Lemmon, "Liveness-enforcing supervision of bounded ordinary Petri nets using partial order methods," *IEEE Trans Automat. Contr.*, vol. 47, no. 7, pp. 1042–1055, 2002.

[22] F.-S. Hsieh, "Reconfigurable fault tolerant deadlock avoidance controller synthesis for assembly production processes," in *Proc. IEEE Internat. Conf. Syst., Man, Cybern.*, 2000, pp. 3045–3050.

[23] A. Giua and C. Seatzu, "Observability of place/transition nets," *IEEE Trans Automat. Contr.*, vol. 47, no. 9, pp. 1424–1437, 2002.

[24] F. Basile, P. Chiacchio, and A. Giua, "On the choice of suboptimal monitor places for supervisory control of Petri nets," in *Proc. IEEE Internat. Conf. Syst., Man, Cybern.*, 1998, pp. 752–757.