2013 European Control Conference (ECC)
July 17-19, 2013, Zürich, Switzerland.

Compact supervisors for general constraint enforcement in Petri net models with uncontrollable transitions

Francesco Basile, Roberto Cordone, and Luigi Piroddi

Abstract— **Petri net (PN) models of production processes are subject to a number of heterogeneous constraints, both static (*e.g.*, limitation and sharing of resources, job limitation) and behavioral (*e.g.*, liveness, reversibility). All of these constraints can be indirectly formulated as generalized mutual exclusion constraints (GMECs), which are conveniently implemented as monitor places suitably connected to the transitions of the open-loop PN plant model. The design procedure is typically sequential, dealing separately with each control objective, possibly resulting in a redundant supervisor. The process is further complicated in the presence of uncontrollable transitions. An integrated modeling approach is here proposed to solve the redundancy problem by accounting for all the specifications in a single design step that optimizes the number of required GMECs and the permissivity of the resulting supervisor. The supervisor can be then implemented as a monitor-based supervisor or, in some cases, as a logical predicate.**

I. Introduction

PETRI nets (PNs) are a convenient formalism for modeling and controlling discrete-event systems (DESs), especially in the presence of a high degree of concurrency and synchronization (see, *e.g.*, [1]). Various specifications (boundedness, mutual exclusion in resource sharing, job limitations, deadlock prevention (DP) or liveness enforcement (LE), etc.) must typically be enforced on an (open-loop) PN model representing the basic structure of the system (*e.g.*, the unconstrained production sequences). The basic control problem is thus that of designing a supervisor that restricts the reachability set of the plant net in closed loop exactly to the set of legal markings $\mathcal{L}$ implicitly defined by these specifications (see, *e.g.*, [2], [3], [4], [5], [6]). Such supervisor is often defined by a set of linear inequalities called Generalized Mutual Exclusion Constraints (GMECs), [2], [7], which are satisfied by the states in $\mathcal{L}$ and violated by any other reachable marking. Such GMECs are easily implemented through places (called *monitors*), suitably connected to the transitions of the PN model of the plant. In more complex cases, a disjunction of GMECs may be necessary to enforce $\mathcal{L}$. In that case, the supervisor is not in the form of a PN anymore, but can be implemented as a logical predicate that can be evaluated on line at a very low cost.

Control specifications can be divided into *static* and *behavioral*, the former being associated directly to individual states, while the latter depend on the structure of the reachability graph of the PN. Bounds on job and resource usage fall in the first category, whereas deadlock prevention (DP), liveness enforcement (LE), reversibility, controllability, etc. are behavioral specifications. This distinction is crucial, since while the implementation of both static and behavioral requirements results in a contraction of the reachable space, only properties of the second category may be lost when some previously reachable states are forbidden. For this reason, constraint enforcement has generally been conceived as a multi-step procedure, where static specifications are implemented first and behavioral requirements (*e.g.*, liveness) afterwards. This ensures that the required behavioral properties are not lost due to the application of further constraints that reduce the reachability set. This approach has some important drawbacks. First of all, the minimality of the supervisor enforcing all the constraints cannot be guaranteed, and some redundancy may be experienced among the different subsets of GMECs. This is generally undesired, since it causes an increase of control code size and of the control response time. The design process is further complicated if multiple behavioral requirements are posed, since the individual enforcement of one such requirement may cause the loss of another one, requiring to iterate the procedure. For example, enforcing liveness after controllability may impair the latter, and *viceversa*.

Regarding the redundancy issue, much research has been devoted specifically to obtaining compact GMEC-based supervisors, focusing exclusively on the DP/LE task (see, *e.g.*, [8], [9], [10]). Several recent contributions have addressed the problem of guaranteeing maximal permissivity with a *minimal* size GMEC-based DP/LE supervisor. These approaches employ an integer linear programming (ILP) formulation of the problem, with the GMEC parameters as decision variables, and suitable constraints enforcing the separation of the legal and illegal marking sets (see, *e.g.*, [11], [12], [13], [14]). This formulation requires the preliminary enumeration of the reachable space, a task which was previously considered unmanageable, but can now be efficiently tackled with recently proposed data structures that provide a compact encoding of the reachable markings [15]. Following the general methodology of [11], [12], an efficient monitor optimization algorithm is proposed in [16], based on a suitable partitioning of the illegal marking set into subsets individually separable with a single GMEC from the whole legal set. This allows to efficiently solve the problem with a systematic exploration of such partitions with a branch & bound (B&B) method. This approach has been recently extended to account for disjunctions of GMECs [17].

This paper addresses the design of the most compact supervisor that enforces *all* the mentioned static and behavioral constraints in a single integrated procedure. This is done by

F. Basile is with the Dipartimento di Ing. dell'Informazione, Ing. Elettrica e Matematica applicata, Università di Salerno, Italy fbasile@unisa.it

R. Cordone is with the Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy roberto.cordone@unimi.it

L. Piroddi is with the Dip. di Elettronica, Informazione e Bioingegneria Politecnico di Milano, Milano, Italy piroddi@elet.polimi.it

applying the approach described in [16], or its extension [17], to a more general classification problem, where the illegal set collects all the markings that violate any of the specifications. To this aim, a generation process is designed to obtain in a single step the legal and illegal marking sets accounting for all the required specifications. The obtained sets are then given as input to the monitor optimizer, which computes the smallest supervisor that enforces all the specifications in a maximally permissive way. The supervisor optimization task is addressed also in the case that the PN has uncontrollable transitions. This requires special care particularly if other behavioral properties (*e.g.*, liveness) are to be enforced besides controllability, as previously mentioned.

II. Preliminaries

A. Petri net basics

A marked PN [18] is a 5-tuple $N = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post}, \boldsymbol{m}_0 \rangle$, where P and T are the (finite and nonempty) sets of n_p places and n_t transitions, with $P \cap T = \emptyset$, $\boldsymbol{Pre}, \boldsymbol{Post} \in \mathbb{N}^{n_p \times n_t}$ are the input and output matrices, and $\boldsymbol{m}_0 \in \mathbb{N}^{n_p}$ is the (initial) marking vector, $\mathbb{N}$ being the set of nonnegative integers. Places (represented as circles) are connected to transitions (represented as bars) through directed weighted arcs. More precisely, $Pre_{k,j}$ [$Post_{k,j}$] represents the weight of an arc going from p_k [t_j] to t_j [p_k] (0 if there is no such arc). In the absence of self-loops, an equivalent information is given by the incidence matrix $\boldsymbol{C} = \boldsymbol{Post} - \boldsymbol{Pre}$. The marking vector $\boldsymbol{m}$ defines the distribution of tokens in places.

A transition $t_j \in T$ is enabled in a marking $\boldsymbol{m}$ (denoted $\boldsymbol{m}[t_j\rangle$) iff $\boldsymbol{m} \geq \boldsymbol{Pre}\,\boldsymbol{e}_j$, where $\boldsymbol{e}_j$ is the jth versor of the $\mathbb{R}^{n_t}$ coordinate space. A transition t_j such that $\boldsymbol{m}[t_j\rangle$ may fire at marking $\boldsymbol{m}$, yielding the marking $\boldsymbol{m}'$ (denoted $\boldsymbol{m}[t_j\rangle\boldsymbol{m}'$), where $\boldsymbol{m}' = \boldsymbol{m} + \boldsymbol{C}\boldsymbol{e}_j$. The reachability set $R(N, \boldsymbol{m}_0)$ collects the markings reachable from $\boldsymbol{m}_0$ by way of enabled transition sequences. The reachability graph is a directed graph $RG = (V, A)$, where $V = R(N, \boldsymbol{m}_0)$ is the set of nodes and $A \subseteq (V \times V)$ the set of arcs, associated to the PN transitions through a labeling function $h : A \to T$. A strongly connected component (SCC) is a maximal subgraph of a directed graph, such that any two of its nodes are connected by a directed path. Let (S, A_S) be an SCC of RG. Then, if $|S| \geq 2$ the PN can evolve indefinitely inside S. (S, A_S) is characterized as *terminal* if there does not exist any $(\boldsymbol{m}_1, \boldsymbol{m}_2) \in A$ with $\boldsymbol{m}_1 \in S$ and $\boldsymbol{m}_2 \in R(N, \boldsymbol{m}_0) \setminus S$.

A place $p_i \in P$ is bounded iff $\exists k > 0$ s.t. $m_i \leq k$, $\forall \boldsymbol{m} \in R(N, \boldsymbol{m}_0)$. A PN is bounded iff all its places are bounded. A transition $t_j \in T$ is live iff $\forall \boldsymbol{m} \in R(N, \boldsymbol{m}_0)$, $\exists \boldsymbol{m}' \in R(N, \boldsymbol{m})$ s.t. $\boldsymbol{m}'[t_j\rangle$. N is live iff all its transitions are live. N is reversible iff $\boldsymbol{m}_0 \in R(N, \boldsymbol{m})$, $\forall \boldsymbol{m} \in R(N, \boldsymbol{m}_0)$. A marking $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$, s.t. $\nexists t_j \in T$ enabled in $\boldsymbol{m}$, is called a dead marking and represents a (total) deadlock state. A PN such that no marking in $R(N, \boldsymbol{m}_0)$ is dead is called deadlock-free. A PN is: i) *deadlock-free* if all the terminal SCCs of RG have cardinality strictly greater than 1, ii) *live* if for any terminal SCC (S, A_S) of RG it holds that $|S| \geq 2$ and $\{t | t = h(a), \forall a \in A_S\} = T$, and iii) *reversible* if RG contains a single SCC [19].

A P-invariant $\boldsymbol{x} \in \mathbb{Z}^{n_p}$ is a left annuler of $\boldsymbol{C}$, *i.e.* such that $\boldsymbol{x}^T\boldsymbol{C} = \boldsymbol{0}$. It satisfies the following marking relation: $\boldsymbol{x}^T\boldsymbol{m} = \boldsymbol{x}^T\boldsymbol{m}_0$, $\forall \boldsymbol{m} \in R(N, \boldsymbol{m}_0)$. The support of a P-invariant $\boldsymbol{x}$ is the set $\| \boldsymbol{x} \| = \{p_i \in P \mid \boldsymbol{x}_i \neq 0\}$. A positive P-invariant is s.t. $\boldsymbol{x} \geq \boldsymbol{0}$. In that case, the set of places $\| \boldsymbol{x} \|$ identifies a conservative component of the PN.

B. GMEC enforcement by means of monitors

A GMEC is a pair $(\boldsymbol{l}, k)$, with $\boldsymbol{l} \in \mathbb{N}^{n_p}$, $k \in \mathbb{N}$, that defines an *admissibility region* $\mathcal{M}(\boldsymbol{l}, k) = \{\boldsymbol{x} \in \mathbb{N}^{n_p} \mid \boldsymbol{l}^T\boldsymbol{x} \leq k\}$. A set of GMECs $(\boldsymbol{L}, \boldsymbol{k})$, with $\boldsymbol{L} = [\boldsymbol{l}_1^T, \boldsymbol{l}_2^T, \ldots, \boldsymbol{l}_{n_c}^T]^T$ and $\boldsymbol{k} = [k_1, k_2, \ldots, k_{n_c}]^T$, defines an admissibility region $\mathcal{M}(\boldsymbol{L}, \boldsymbol{k}) = \cap_{i=1}^{n_c} \mathcal{M}(\boldsymbol{l}_i, k_i)$. Provided $\boldsymbol{m}_0 \in \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$ holds, a control sub-net consisting of n_c additional places (monitors) connected to the PN transitions through the incidence matrix $\boldsymbol{C}_C = -\boldsymbol{L}\boldsymbol{C}$ and marked according to $\boldsymbol{m}_{C0} = \boldsymbol{k} - \boldsymbol{L}\boldsymbol{m}_0$ enforces the said constraints [2], [7]. The controlled PN has n_c new P-invariants, and the designed controller is maximally permissive, *i.e.* it prevents only transition firings that would result in a GMEC violation.

C. GMEC optimization as a classification problem

A set of GMECs $(\boldsymbol{L}, \boldsymbol{k})$ can be envisaged as a *linear classifier*, separating the markings in $\mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$ from those outside. Conditions for the existence of a linear classifier $(\boldsymbol{L}, \boldsymbol{k})$ that separates any two given (disjoint) marking sets, $\mathcal{L}$ (the *legal* set) and $\mathcal{U}$ (the *illegal* set), are discussed in [17].

Theorem 1: [17] There exists a linear classifier for marking sets $\mathcal{L}$ and $\mathcal{U}$ *iff* there does not exist a marking $\boldsymbol{m} \in \mathcal{U}$ such that $\boldsymbol{m} \in P_{\mathcal{L}}$, where $P_{\mathcal{L}}$ is the convex hull.

The design of a linear classifier can be interpreted as the search for an optimal covering of the illegal set $\mathcal{U}$ with suitable subsets $\mathcal{U}_i$, $i = 1, \ldots, n_c$, such that for each subset there exists a GMEC that separates it from $\mathcal{L}$. Indeed, the resulting set of GMECs provides a linear classifier that separates $\mathcal{U}$ from $\mathcal{L}$. This set covering problem is optimally solved by means of the B&B method explained in [16].

If the condition of Thm. 1 is violated, $\mathcal{U}$ can be separated from $\mathcal{L}$ by means of a nonlinear classifier, obtained as a disjunction of linear classifiers, $(\boldsymbol{L}_j, \boldsymbol{k}_j)$, $j = 1, \ldots, n_P$ [17]. Such classifier includes all the markings $\boldsymbol{m}$ such that $\boldsymbol{L}_j\boldsymbol{m} \leq \boldsymbol{k}_j$, for at least one j. In other words, it defines the admissibility region $\cup_{j=1}^{n_P} \mathcal{M}(\boldsymbol{L}_j, \boldsymbol{k}_j)$. The separation is achieved if the mentioned region includes $\mathcal{L}$ and has an empty intersection with $\mathcal{U}$. This is equivalent to partitioning the set $\mathcal{L}$ in subsets $\mathcal{L}_j$, $j = 1, \ldots, n_P$, such that $(\boldsymbol{L}_j, \boldsymbol{k}_j)$ separates the whole $\mathcal{U}$ from $\mathcal{L}_j$.

A supervisor can be defined based on this disjunction of classifiers by means of a logical predicate, that must be evaluated to establish the enabling of a transition. A logical predicate is a function $[P](\boldsymbol{m}, t) : (\boldsymbol{m}, t) \to \{0, 1\}$ where $t \in T$. If $[P](\boldsymbol{m}, t) = 1$ the transition is enabled under the marking $\boldsymbol{m}$. The predicate $[P](\boldsymbol{m}, t) = \bigvee_j \boldsymbol{L}_j{}^T(\boldsymbol{m} + \boldsymbol{C}(:, t)) \leq k_j$ implements $\mathcal{L}$. Evaluating this predicate typically requires a negligible time compared to the usual time-scale involved in DES applications. The resulting supervisor is said to be *interpreted*, since it is not modeled by a PN.

The quest for such a nonlinear classifier can be envisaged as the solution of a two-level set covering problem, operating on $\mathcal{L}$ at the upper level, and on $\mathcal{U}$ at the lower one. An extended version of the B&B approach of [16] has been developed in [17] to deal with this more complex problem.

III. Monitor synthesis with uncontrollable transitions

Consider now the problem of restricting the reachability set of a PN within $\mathcal{L}$ in the presence of uncontrollable transitions. The set of transitions T is partitioned into two disjoint subsets, T_c and T_u, denoting the controllable and uncontrollable transitions (associated to controllable and uncontrollable events), respectively.

Definition 1: Consider a net N with $T_c \neq \emptyset$. We define the uncontrollable sub-net of N, denoted as $N_u = \langle P, T_u, \mathbf{Pre}_u, \mathbf{Post}_u \rangle$, the sub-net obtained from N eliminating every controllable transition.

It is immediate to see that $R(N_u, \boldsymbol{m}) \subseteq R(N, \boldsymbol{m})$.

Definition 2: A legal marking set $\mathcal{L} \subseteq \mathbb{N}^{n_p}$ is *behaviorally controllable* w.r.t. a PN N with initial marking $\boldsymbol{m}_0$ if $\cup_{\boldsymbol{m} \in \mathcal{L}} R(N_u, \boldsymbol{m}) \subseteq \mathcal{L}$, where N_u is the uncontrollable sub-net of N. ■

According to this definition, $\mathcal{L}$ is controllable if no forbidden marking is reachable from any marking $\boldsymbol{m} \in \mathcal{L}$ by firing a sequence containing only uncontrollable transitions.

When the controller is modeled by a PN, it is possible to disable a transition t only if there is an arc from a control place to t and the control place is insufficiently marked. Therefore, a simple, but possibly restrictive [20] condition that ensures controllability is to avoid altogether arcs directed from monitor places to uncontrollable transitions.

Definition 3: [21] Let N be a PN with transition set $T = T_c \cup T_u$ with $T_c \cap T_u = \emptyset$, and denote as $\boldsymbol{C}_u$ the incidence matrix of the uncontrollable sub-net N_u. Let also $\mathcal{L}$ be a set of legal markings and $(\boldsymbol{L}', k')$ a set of GMECs such that (i) $R(N_c, \boldsymbol{m}_0) = \mathcal{L}$, where N_c denotes the controlled net resulting from the monitor implementation of $(\boldsymbol{L}', k')$, and (ii) $\boldsymbol{L}'\boldsymbol{C}_u \leq \boldsymbol{0}$. Then, $\mathcal{L}$ is said to be *structurally controllable*. ■

In the sequel, unless otherwise stated, we will refer exclusively to the concept of behavioral controllability, dropping the attribute "behavioral" for brevity.

Now, to reduce a non-controllable specification $\mathcal{L}_{\bar{c}}$ to a controllable one $\mathcal{L}_c$, one must avoid reaching the set of markings $\mathcal{L}_{uc} = \{\boldsymbol{m} \in \mathcal{L}_{\bar{c}} \mid \boldsymbol{m}[\sigma > \boldsymbol{m}',\ \boldsymbol{m}' \notin \mathcal{L}_{\bar{c}}, \sigma \in T_u^*\}$. The class $\Omega(\mathcal{L}_{\bar{c}}) = \{\mathcal{K} \subseteq \mathcal{L}_{\bar{c}} \mid \mathcal{K} \text{ is controllable}\}$ of controllable subsets of $\mathcal{L}$ admits a unique and nonempty maximal controllable subset [22]. Indeed, the element $\mathcal{L}_c = \sup \Omega(\mathcal{L}_{\bar{c}}) = \mathcal{L}_{\bar{c}} \setminus \mathcal{L}_{uc}$, called *supremal controllable subset*, provides an upper bound to the set of legal markings.

Unfortunately, $\mathcal{L}_c$ may not be fully characterizable in terms of a plain set of GMECs, even if $\mathcal{L}_{\bar{c}}$ is. Let $\Omega_{n_c}(\mathcal{L}_{\bar{c}}) = \{\mathcal{K} \subseteq \mathcal{L}_{\bar{c}} \mid \exists \boldsymbol{L}'' \in \mathbb{Z}^{n_c \times n_p}, \boldsymbol{k}'' \in \mathbb{Z}^{n_c} : R(N_c, \boldsymbol{m}_0) = \mathcal{K}, \mathcal{K} \text{ is controllable}\}$, where N_c is the controlled net resulting from the monitor implementation of $(\boldsymbol{L}'', k'')$. $\Omega_{n_c}(\mathcal{L}_{\bar{c}})$ represents the set of *controllable* subsets of $\mathcal{L}_{\bar{c}}$ that are *realizable by a set of n_c linear inequalities*. The class $\Omega_{n_c}(\mathcal{L}_{\bar{c}})$ is not closed under union and not empty [23], so that a maximal element exists but it is not necessarily unique. By enforcing a specific (maximal) controllable subset $\mathcal{K}_j = \mathcal{M}(\boldsymbol{L}_j, \boldsymbol{k}_j)$ of $\mathcal{L}_{\bar{c}}$ belonging to $\Omega_{n_c}(\mathcal{L}_{\bar{c}})$, one may prevent the closed loop system from reaching some perfectly legal markings, contained in $\bar{\mathcal{K}} \setminus \mathcal{K}_j$, where $\bar{\mathcal{K}} = \cup_{\mathcal{K} \in \Omega_{n_c}(\mathcal{L}_{\bar{c}})} \mathcal{K}$. Set $\bar{\mathcal{K}}$ is not generally representable as a set of GMECs, but can be realized as a *disjunction* of the sets of GMECs enforcing the elements $\mathcal{K}_j$.

As already recalled, the problem may be further complicated if other behavioral properties (*e.g.*, liveness) are required, besides controllability.

IV. The proposed method

A. Constraint enforcement as a classification problem

Let N be a (possibly unbounded) PN with initial marking $\boldsymbol{m}_0$, and with transition set $T = T_c \cup T_u$ with $T_c \cap T_u = \emptyset$. Assume also that the set of GMECs $(\boldsymbol{L}, \boldsymbol{k})$ encompasses all required static specifications. In the following we develop an integrated procedure to determine the smallest (in terms of the number of involved GMECs) maximally permissive supervisor that guarantees liveness and controllability, as well as the aforementioned specifications[1]. This procedure consists of two steps:

1) Determine the sets of legal and illegal markings, denoted respectively $\mathcal{L}$ and $\mathcal{U}$. Set $\mathcal{L}$ is defined so that:
   (i) if the PN evolution is confined in $\mathcal{L}$ all constraints defined by $(\boldsymbol{L}, \boldsymbol{k})$ are respected,
   (ii) $\mathcal{L}$ is included in a controllable subset of $\mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$,
   (iii) the PN can indefinitely evolve inside $\mathcal{L}$, and
   (iv) in doing so, all PN transitions can be enabled infinitely many times.
2) Determine a logical disjunction of linear classifiers $(\boldsymbol{L}'_j, \boldsymbol{k}'_j)$ that separates $\mathcal{U}$ from $\mathcal{L}$, *i.e.* such that $\mathcal{L} \subseteq \cup_j \mathcal{M}(\boldsymbol{L}'_j, \boldsymbol{k}'_j)$ and $\mathcal{M}(\boldsymbol{L}'_j, \boldsymbol{k}'_j) \cap \mathcal{U} = \emptyset$ for all j, and construct the corresponding logical-predicate-based supervisor.

Set $\mathcal{U}$ needs to contain only the *boundary* illegal markings (*i.e.* those adjacent to the legal ones), since by forbidding them no subsequent illegal marking will ever be reached. Notice that, while Step (2) is satisfactorily solved using the method introduced in [17], a method for addressing Step (1) still needs to be developed, as discussed in the sequel.

B. Generation of the legal and illegal sets

The next theorem provides a way to characterize the set of legal markings $\mathcal{L}$ for a net on which liveness and controllability must be enforced together with static constraints described by a set of GMECs $(\boldsymbol{L}, \boldsymbol{k})$. Other behavioral properties, such as DP or reversibility, are discussed later.

Theorem 2: Consider a PN N with initial marking $\boldsymbol{m}_0$ and transition set $T = T_c \cup T_u$, with $T_c \cap T_u = \emptyset$. Let also $RG = (V, A)$ be the reachability graph of the net, with $V = R(N, \boldsymbol{m}_0)$, and $(\boldsymbol{L}, \boldsymbol{k})$ be a set of GMECs, such that $R(N, \boldsymbol{m}_0) \cap \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$ is a finite set and $\boldsymbol{m}_0 \in \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$. Finally, let $V^\star$ be the set of all markings $\boldsymbol{m}$ such that:

(i) $\boldsymbol{m} = \boldsymbol{m}_0$ or there exists in RG a directed path from $\boldsymbol{m}_0$ to $\boldsymbol{m}$ whose nodes all belong to $R(N, \boldsymbol{m}_0) \cap \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$;
(ii) $\forall t_j \in T$ there exist $\boldsymbol{m}', \boldsymbol{m}'' \in V^\star$ such that (a) $(\boldsymbol{m}', \boldsymbol{m}'') \in A$, (b) $h(\boldsymbol{m}', \boldsymbol{m}'') = t_j$, and (c) there exists in RG a directed path from $\boldsymbol{m}$ to $\boldsymbol{m}'$ whose nodes all belong to $R(N, \boldsymbol{m}_0) \cap \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$;

[1] Notice that the GMECs of the resulting supervisor are not necessarily related to $(\boldsymbol{L}, \boldsymbol{k})$.

(iii) $\forall \boldsymbol{m}' \notin V^\star$ such that $(\boldsymbol{m}, \boldsymbol{m}') \in A$, $h(\boldsymbol{m}, \boldsymbol{m}') \in T_c$.

If $V^\star$ is nonempty, then a supervisor that allows all markings in the legal set $\mathcal{L}_c = V^\star$, while forbidding all other reachable markings, is a maximally permissive supervisor that enforces the maximal subset of reachable markings controllable with respect to $(\boldsymbol{L}, \boldsymbol{k})$, and the liveness of the controlled PN. ■

Proof: Let N_c be the controlled PN. Then, by assumption, $R(N_c, \boldsymbol{m}_0) \subseteq V^\star$, since the supervisor forbids all reachable markings not belonging to $V^\star$. Conversely, by condition (i), $V^\star$ contains only markings that are reachable from $\boldsymbol{m}_0$ through evolution paths involving states belonging to $V^\star$. Therefore, $R(N_c, \boldsymbol{m}_0) = V^\star$. Since the supervisor enforces a forbidden-state policy, the reachability graph of N_c is the subgraph induced by $V^\star$ on RG, denoted $RG^\star = (V^\star, A^\star)$, where $A^\star = \{(\boldsymbol{m}', \boldsymbol{m}'') \in A,\ s.t.\ \boldsymbol{m}', \boldsymbol{m}'' \in V^\star\}$. Condition (i) also guarantees that N_c respects all the requirements expressed by the set of GMECs $(\boldsymbol{L}, \boldsymbol{k})$ and that $R(N_c, \boldsymbol{m}_0)$ is a finite set, since it implies that $R(N_c, \boldsymbol{m}_0) \subseteq \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$. Condition (ii) guarantees the liveness of N_c, because it ensures that the state evolution on graph $RG^\star$ cannot end in single (deadlock) states or in state cycles not involving all transitions of the PN. Condition (iii) imposes that all exits from set $V^\star$ must result from state transitions through arcs labeled with controllable transitions only. Conversely, all the markings reachable by firing uncontrollable transitions are legal. Since $R(N_c, \boldsymbol{m}_0) = V^\star$, where N_c is the controlled PN, this implies that $\mathcal{L}_c$ is controllable. Finally, the supervisor is maximally permissive, since $V^\star$ includes all the markings which satisfy all conditions. ■

Notice that a PN may not admit a subgraph of RG with the requirements (i-iii) of Thm. 2, meaning that liveness and controllability cannot be enforced on the PN together with the constraints $(\boldsymbol{L}, \boldsymbol{k})$.

Thm. 2 can be adapted to account for different behavioral requirements. If liveness is not required and it is only necessary that N_c be deadlock-free, Condition (ii) can be relaxed as follows:

(ii′) *there exists* $\boldsymbol{m}' \in V^\star \setminus \{\boldsymbol{m}\}$ *such that* $(\boldsymbol{m}, \boldsymbol{m}') \in A$.

On the other hand, if reversibility is also required, an additional condition must be included, namely:

(iv) $\boldsymbol{m} = \boldsymbol{m}_0$ or there exists in RG a directed path from $\boldsymbol{m}$ to $\boldsymbol{m}_0$ whose nodes all belong to $R(N, \boldsymbol{m}_0) \cap \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$;

The controllable legal set can be determined through the application of the following Algorithm 1.

Algorithm 1:

1) Construct a *truncated* reachability graph $RG' = (V', A')$, starting with $V' = \{\boldsymbol{m}_0\}$ and $A' = \emptyset$, and iteratively augmenting the graph as long as possible with the following rule: for all $\boldsymbol{m} \in V'$ and all $t \in T$ such that $\boldsymbol{m}[t > \boldsymbol{m}'$ and $\boldsymbol{m}' \in \mathcal{M}(\boldsymbol{L}, \boldsymbol{k}) \setminus V'$, set $V' = V' \cup \{\boldsymbol{m}'\}$ and $A' = A' \cup \{(\boldsymbol{m}, \boldsymbol{m}')\}$.
2) Recursively prune RG' of:
   (i) Any terminal SCC (S, A_S) with $S \subseteq V'$, $A_S = \{(\boldsymbol{m}, \boldsymbol{m}') \in A' \mid \boldsymbol{m}, \boldsymbol{m}' \in S\}$, such that $\{t_j \in T \mid t_j = h(a), a \in A_S\} \subset T$;
   (ii) Any state $\boldsymbol{m}$ such that $\exists a = (\boldsymbol{m}, \boldsymbol{m}') \in A$, with $\boldsymbol{m}' \notin V'$ and $h(a) \in T_u$.

If liveness is not required, the pruning task (2.i) can be restricted to terminal SCCs of cardinality 1 (deadlocks). On the other hand, if reversibility is required as well, RG' must be reduced to its SCC containing $\boldsymbol{m}_0$ (see Section II-A).

The following theorem states that Algorithm 1 indeed provides the correct legal set as of Thm. 2, *i.e.* $RG' = RG^\star$.

Theorem 3: Consider a PN N with initial marking $\boldsymbol{m}_0$ and transition set $T = T_c \cup T_u$, with $T_c \cap T_u = \emptyset$. Let also $(\boldsymbol{L}, \boldsymbol{k})$ be a set of GMECs, such that $R(N, \boldsymbol{m}_0) \cap \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$ is a finite set that includes $\boldsymbol{m}_0$. Let $RG = (V, A)$ be the reachability graph of the net, with $V = R(N, \boldsymbol{m}_0)$ and assume that there exists a nonempty $V^\star \subseteq V$ satisfying Thm. 2. Finally, let $RG' = (V', A')$ be the graph resulting from the application of Algorithm 1 to N. Then $RG' = RG^\star$, where $RG^\star = (V^\star, A^\star)$ is the subgraph induced by $V^\star$ on RG. ■

Proof: By construction, RG' is a subgraph of $R(N, \boldsymbol{m}_0)$. Step (1) of Algorithm 1 ensures that $\boldsymbol{m}_0 \in V'$ and that $V' \subseteq \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$ as required by condition (i) of Thm. 2 (notice that $\boldsymbol{m}_0 \in \mathcal{M}(\boldsymbol{L}, \boldsymbol{k})$ by hypothesis). The constructive process of Step (1) of the algorithm ensures that all markings in V' are reachable from $\boldsymbol{m}_0$ by way of state evolution paths including only markings in V' itself. This property is not lost after the application of Step (2), which recursively prunes RG' of *terminal* SCCs that jeopardize liveness. Therefore, condition (i) of Thm. 2 still holds. The pruning step (2) ensures that the state evolution on graph RG' cannot end in terminal SCCs not involving all transitions of the PN. This implies condition (ii) of Thm. 2. The pruning step eliminates also states that have a successor outside V' reached by firing an uncontrollable transition, implementing condition (iii) of Theorem 2. Since the pruning step (2) of Algorithm 1 only eliminates components that violate conditions (ii-iii), RG' is the *maximal* subgraph of RG that abides by conditions (i-iii) of Thm. 2 and therefore coincides with $RG^\star$. ■

Note that the building of $V^\star$ does not require the computation of $R(N, \boldsymbol{m}_0)$. Moreover, given $\mathcal{L}$, the set of boundary illegal markings is trivially computed as $\mathcal{U} = \mathcal{L}_{suc} \setminus \mathcal{L}$, where $\mathcal{L}_{suc} = \{\boldsymbol{m}' \in R(N, \boldsymbol{m}_0) \mid (\boldsymbol{m}, \boldsymbol{m}') \in A \wedge \boldsymbol{m} \in V^\star\}$. In other words, $\mathcal{L}_{suc}$ is the set of illegal markings (outside $V^\star$) that can be reached from legal markings (in $V^\star$) by firing a single transition. Notice that while $R(N, \boldsymbol{m}_0) \setminus \mathcal{L}$ can be unbounded, the successor marking set $\mathcal{L}_{suc}$ is finite, given that both $\mathcal{L}$ and T are finite sets. Consequently, the set $\mathcal{U}$ is also necessarily finite. This is particularly relevant, given that $\mathcal{L}$ and $\mathcal{U}$ need to be enumerated to build the ILP formulation that describes the optimal supervisor.

C. Design of the optimal supervisor

Once the sets $\mathcal{L}$ and $\mathcal{U}$ have been determined the optimal supervisor must be calculated. Depending on the structure of the two sets one of the two algorithms mentioned in Section II-C applies. More precisely, one has first to ascertain if there exists a marking $\boldsymbol{m} \in \mathcal{U}$ such that $\boldsymbol{m} \in P_{\mathcal{L}}$, where $P_{\mathcal{L}}$ is the convex hull of $\mathcal{L}$. To this end, consider the following system of linear inequalities in $(\boldsymbol{l}, k)$:

$$\boldsymbol{l} \cdot \boldsymbol{x} \leq k \qquad \boldsymbol{x} \in \mathcal{L} \tag{1a}$$

$$\boldsymbol{l} \cdot \boldsymbol{u} \geq k + 1 \tag{1b}$$

State $\boldsymbol{u}$ belongs to $P_{\mathcal{L}}$ if system (1) does not admit feasible solutions [17]. Repeating the test for all markings in $\mathcal{U}$ yields the required information.

If it turns out that $\mathcal{U} \cap P_{\mathcal{L}} = \emptyset$, then the simpler algorithm of [16] can be applied and the resulting supervisor implemented as a PN as well, introducing the necessary monitors. However, especially in the partially controllable case the condition required by Thm. 1 is often violated, as discussed in Section III, and a disjunctive linear classifier must be employed. Such nonlinear classifier can be computed in two ways. The first option is to implement separately the linear classifiers $(\boldsymbol{L}_j, \boldsymbol{k}_j)$ that enforce each maximal controllable subset $\mathcal{K}_j$ realizable by GMECs, ensuring controllability (each linear classifier will actually enforce $V_j^\star \subseteq \mathcal{M}(\boldsymbol{L}_j, \boldsymbol{k}_j)$), and finally constructing the interpreted supervisor based on the disjunctive logical predicate $[P](\boldsymbol{m}, t) = \bigvee_j \boldsymbol{L}_j{}^T(\boldsymbol{m}+\boldsymbol{C}(:,t)) \leq k_j$. This approach implies a repeated application of the procedure of [16]. Alternatively, one can directly apply the procedure of [17] to find the optimal nonlinear classifier enforcing $\mathcal{L}$. While more complex, this second approach may provide a more permissive solution, since $\bigcup_j V_j^\star \subseteq \mathcal{L}$.

The complexity of the classification problem increases rapidly with the size of $\mathcal{L}$ and $\mathcal{U}$. However, if the search is restricted to linear classifiers with nonnegative coefficients, it can be shown that $\mathcal{L}$ and $\mathcal{U}$ can be thinned to their maximal and minimal components, respectively (see [11], [12], [13], [14], [17]), which has often a huge impact in practice. Besides, this sign restriction is not necessarily a limitation, since the envisaged constraints more often require the imposition of conservative components in the PN, which are obtained through GMECs with nonnegative coefficients.

V. Simulation example

Consider the PN in Fig. 1, that represents a process with two production sequences. The PN has 3 uncontrollable transitions, namely t_2, t_5, and t_9. The two production sequences

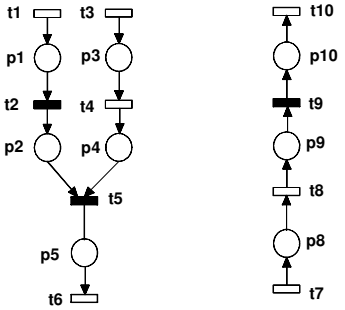

Fig. 1. Example 1: Open-loop plant.

operate with two shared resources R_1 and R_2, with the following constraints, respectively:

$$\boldsymbol{m}(p_3) + \boldsymbol{m}(p_4) + \boldsymbol{m}(p_{10}) \leq 1 \quad (2)$$
$$\boldsymbol{m}(p_5) + \boldsymbol{m}(p_8) \leq 2 \quad (3)$$

The production sequences are also subject to additional bounds related to the maximum number of jobs that can be activated concurrently, namely:

$$\boldsymbol{m}(p_1) + \boldsymbol{m}(p_2) + \boldsymbol{m}(p_5) \leq 3 \quad (4)$$
$$\boldsymbol{m}(p_3) + \boldsymbol{m}(p_4) + \boldsymbol{m}(p_5) \leq 3 \quad (5)$$
$$\boldsymbol{m}(p_8) + \boldsymbol{m}(p_9) + \boldsymbol{m}(p_{10}) \leq 3 \quad (6)$$

The aim is to design a minimal supervisor, capable of imposing constraints (2-6) and enforcing the liveness of the controlled PN in a maximally permissive way, notwithstanding the presence of the uncontrollable transitions.

A direct implementation of the two GMECs associated to the two resources would yield uncontrollable constraints, due to t_9 and t_5, respectively. A standard approach would be to reformulate the two constraints to make them more conservative but structurally controllable. Following [23], constraint (2) admits only one reformulation, namely:

$$\boldsymbol{m}(p_3) + \boldsymbol{m}(p_9) + \boldsymbol{m}(p_{10}) \leq 1, \quad (7)$$

whereas constraint (3) can be transformed either to

$$\boldsymbol{m}(p_1) + \boldsymbol{m}(p_2) + \boldsymbol{m}(p_5) + \boldsymbol{m}(p_8) \leq 2 \text{ or} \quad (8)$$
$$\boldsymbol{m}(p_4) + \boldsymbol{m}(p_5) + \boldsymbol{m}(p_8) \leq 2. \quad (9)$$

A direct implementation of the transformed constraints (4-8) would result in a live PN, whereas a non-live PN would be obtained with constraint (9) in place of (8).

A first supervisor can be obtained by calculating a linear classifier for each of the two cases of the reformulated constraints, and then composing the two sets of GMECs in a disjunctive form. Consider first the case with the constraints (4-8). The first task is to generate the set of safe states, according to the method explained in Section IV-B. As a result, 201 safe states are obtained, and 404 unsafe ones. Then the direct monitor optimization method of [16] finds that 2 GMECs are sufficient to respect all the given constraints and yield a live controlled PN with 201 states (see Fig. 2):

$$5\boldsymbol{m}(p_3) + \boldsymbol{m}(p_4) + \boldsymbol{m}(p_5) + 4\boldsymbol{m}(p_9) + 4\boldsymbol{m}(p_{10}) \leq 7 \quad (10)$$
$$\boldsymbol{m}(p_1) + \boldsymbol{m}(p_2) + \boldsymbol{m}(p_5) + \boldsymbol{m}(p_8) \leq 2 \quad (11)$$

The second constraint is actually equal to (8), while the

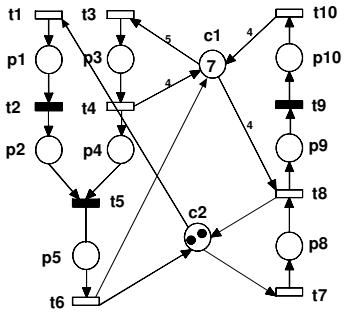

Fig. 2. Optimally controlled PN for constraints (4-8).

first is a linear combination of (7) and (5). In fact, it implies that $\boldsymbol{m}(p_3) + \boldsymbol{m}(p_9) + \boldsymbol{m}(p_{10}) \leq 7/4$, which is equivalent to (7). Also, by construction, one cannot mark $\{p_3, p_4, p_5\}$ with more than 3 tokens overall, which is equivalent to (5). Constraints (4) and (6) are also respected.

Consider now the second reformulation, which amounts to the constraints (4-7), and (9). In this case, 276 safe and 570 unsafe states are obtained, respectively. Three GMECs are necessary to separate them:

$$\boldsymbol{m}(p_3) + \boldsymbol{m}(p_9) + \boldsymbol{m}(p_{10}) \leq 1 \quad (12)$$
$$\boldsymbol{m}(p_3) + 3\boldsymbol{m}(p_4) + 3\boldsymbol{m}(p_5) + 4\boldsymbol{m}(p_8) \leq 8 \quad (13)$$
$$\boldsymbol{m}(p_1) + \boldsymbol{m}(p_2) + \boldsymbol{m}(p_5) \leq 3 \quad (14)$$

as shown in Fig. 3. Constraints (12) and (14) are identical to

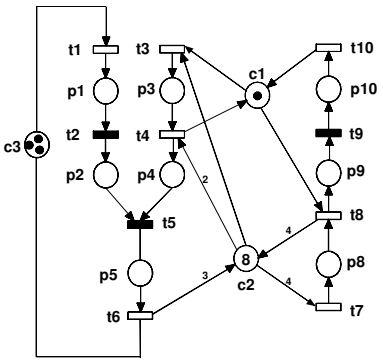

Fig. 3. Optimally controlled PN for constraints (4-7) and (9).

(7) and (4), respectively, while (13) is a linear combination of $\boldsymbol{m}(p_3) + \boldsymbol{m}(p_8) \leq 1$, which is required for the liveness of the controlled PN, and constraint (9). With the 3 GMECs inserted, constraints (5) and (6) are automatically verified.

Using a disjunction of the two controllers one obtains a reachable set with 365 states, corresponding to the union of the two individual legal sets, using a total of 5 GMECs.

On the other hand, the application of Algorithm 1 to the problem, to calculate the maximal legal set compatible with all the constraints, *i.e.* (2-6), and guaranteeing both liveness and controllability, yields a safe set with 384 states, proving the previous solutions to be sub-optimal. 851 boundary unsafe states are also obtained. Now, applying the optimization method of [17], one obtains that the disjunction of the following two sets of GMECs separates the safe and unsafe sets:

$$5\boldsymbol{m}(p_1) + 5\boldsymbol{m}(p_2) + \boldsymbol{m}(p_4) + 6\boldsymbol{m}(p_5) + 7\boldsymbol{m}(p_8) \leq 17 \quad (15)$$
$$5\boldsymbol{m}(p_3) + \boldsymbol{m}(p_4) + \boldsymbol{m}(p_5) + 4\boldsymbol{m}(p_9) + 4\boldsymbol{m}(p_{10}) \leq 7 \quad (16)$$

and

$$\boldsymbol{m}(p_3) + 3\boldsymbol{m}(p_4) + 3\boldsymbol{m}(p_5) + 4\boldsymbol{m}(p_8) \leq 8 \quad (17)$$
$$\boldsymbol{m}(p_3) + \boldsymbol{m}(p_9) + \boldsymbol{m}(p_{10}) \leq 1 \quad (18)$$
$$\boldsymbol{m}(p_1) + \boldsymbol{m}(p_2) + \boldsymbol{m}(p_5) \leq 3 \quad (19)$$

This solution has the same complexity (in terms of the overall number of GMECs required) of the previously derived one, but is more permissive.

VI. Conclusions

A direct monitor optimization approach has been here applied to the problem of enforcing generalized mutual exclusion constraints accounting for various types of static and behavioral specifications and extended to the case of uncontrollable transitions. The overall design process produces in a single step a minimum size maximally permissive supervisor that can be implemented by monitor places or efficient logical predicates, corresponding to the disjunction of various sets of GMECs. The method requires the calculation of the set of controllable legal markings resulting from the application of all the mentioned constraints and requirements. Then, the corresponding set of boundary illegal markings is obtained and a classification problem is formulated which aims at the separation of the two sets. An efficient solution method is employed based on a B&B approach.

References

[1] M. Zhou and K. Venkatesh, *Modelling, Simulation and Control of Flexible Manufacturing Systems: A Petri Net Approach*. Singapore: World Scientific, 1998.

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

[3] J. Moody, M. Lemmon, and P. Antsaklis, "Supervisory control of Petri nets with uncontrollable/unobservable transitions," in 35^{th} *IEEE Conf. on Decision and Control*, vol. 4, Kobe, Japan, December 1996, pp. 4433–4438.

[4] Y. Li and W. Wonham, "Control of vector discrete-event systems II - controller synthesis," *IEEE Trans. on Automatic Control*, vol. 39, no. 3, pp. 512–531, March 1994.

[5] B. Krogh and L. Holloway, "Synthesis of feedback control logic for discrete manufacturing systems," *Automatica*, vol. 27, no. 4, pp. 641–651, July 1991.

[6] F. Basile, P. Chiacchio, and C. Carbone, "Feedback control logic for backward conflict free choice nets," *IEEE Trans. on Automatic Control*, vol. 52, no. 3, pp. 387–400, March 2007.

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

[8] M. Uzam, Z. Li, and M. Zhou, "Identification and elimination of redundant control places in Petri net based liveness enforcing supervisors of FMS," *Int. J. Adv. Manuf. Technol.*, vol. 35, no. 1-2, pp. 150–168, 2007.

[9] L. Piroddi, R. Cordone, and I. Fumagalli, "Selective siphon control for deadlock prevention in Petri nets," *IEEE Trans. Syst. Man Cybern. Part A, Syst. Humans*, vol. 38, no. 6, pp. 1337–1348, 2008.

[10] ——, "Combined siphon and marking generation for deadlock prevention in Petri nets," *IEEE Trans. Syst. Man Cybern. Part A, Syst. Humans*, vol. 39, no. 3, pp. 650–661, 2009.

[11] A. Nazeem, S. Reveliotis, Y. Wang, and S. Lafortune, "Optimal deadlock avoidance for complex resource allocation systems through classification theory," in *Proc.* 10^{th} *IFAC Int. Workshop on Discrete Event Systems*, Berlin, Germany, 2010, pp. 277–284.

[12] ——, "Designing compact and maximally permissive deadlock avoidance policies for complex resource allocation systems through classification theory: The linear case," *IEEE Trans. Autom. Control*, vol. 56, no. 8, pp. 1818–1833, 2011.

[13] Y. Chen, Z. Li, M. Khalgui, and O. Mosbahi, "Design of a maximally permissive liveness-enforcing Petri net supervisor for flexible manufacturing systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 2, pp. 374–393, 2011.

[14] Y. Chen and Z. Li, "Design of a maximally permissive liveness-enforcing supervisor with a compressed supervisory structure for flexible manufacturing systems," *Automatica*, vol. 47, pp. 1028–1034, 2011.

[15] A. Nazeem and S. Reveliotis, "A practical approach to the design of maximally permissive liveness enforcing supervisors for complex resource allocation systems," in *Proc.* 6^{th} *IEEE Conf. on Automation Science and Engineering*, Toronto, Canada, 2010, pp. 451–458.

[16] R. Cordone and L. Piroddi, "Parsimonious monitor control of Petri net models of FMS," *IEEE Trans. Syst. Man Cybern., Part A Syst. Humans*, 2012, [to appear].

[17] R. Cordone, A. Nazeem, L. Piroddi, and S. Reveliotis, "Maximally permissive deadlock avoidance for sequential resource allocation systems using disjunctions of linear classifiers," in *Proc.* 51^{st} *IEEE Conf. on Decision and Control*, Maui (Hawaii), USA, 2012.

[18] T. Murata, "Petri nets: properties, analysis and application," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[19] I. Fumagalli, L. Piroddi, and R. Cordone, "A reachability graph partitioning technique for the analysis of deadlock prevention methods in bounded Petri nets," in *Proc. American Control Conference, ACC2010*, Baltimore (MD), USA, 2010, pp. 3365–3370.

[20] A. Ghaffari, N. Rezg, and X. Xie, "Design of a live and maximally permissive petri net controller using the theory of regions," *IEEE Trans. On Robotics and Automation*, vol. 19, no. 1, pp. 137–141, February 2003.

[21] J. Moody and P. Antsaklis, "Petri net supervisors for DES with uncontrollable and unobservable transitions," *IEEE Trans. on Aut. Control*, vol. 45, no. 3, pp. 462–476, March 2000.

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

[23] F. Basile, P. Chiacchio, and A. Giua, "Suboptimal supervisory control of Petri nets in presence of uncontrollable transitions via monitor places," *Automatica*, vol. 42, pp. 995–1004, 2006.