Supervisory Control of Deterministic Petri Nets with Regular Specification Languages *†

Ratnesh Kumar       Lawrence E. Holloway

Department of Electrical Engineering
University of Kentucky
Lexington, KY 40506-0046

Abstract

Algorithms for computing a minimally restrictive control in the context of supervisory control of discrete event systems have been well developed when both the plant and the desired behavior are given as regular languages. In this paper we extend such prior results by presenting an algorithm for computing a minimally restrictive control when the plant behavior is a deterministic Petri net language and the desired behavior is a regular language. As part of the development of the algorithm, we establish the following results that are of independent interest: (i) The problem of determining whether a given deterministic Petri net language is controllable with respect to another deterministic Petri net language is reducible to a reachability problem of Petri nets. (ii) The problem of synthesizing the minimally restrictive supervisor so that the controlled system generates the supremal controllable sublanguage is reducible to a forbidden marking problem. In particular, we can directly identify the set of forbidden markings without having to construct any reachability tree.

Keywords: discrete event systems, Petri nets, supervisory control

1 Introduction

Supervisory control theory of discrete event systems (DESs) was initiated by Ramadge and Wonham [21] to study control of qualitative behavior of systems whose behavior evolve according to the occurrences of events; examples of which include manufacturing systems, communication networks, database management systems, etc. It is well known that given an

*An earlier version of this paper appeared in [14].
†This research was supported in part by the Center for Robotics and Manufacturing at the University of Kentucky, NSF grants NSF-ECS-9308737 and NSF-ECS-9409712, NASA grant NGT-40049, and Rockwell International.
uncontrolled plant with behavior $L$ and a desired behavior $K \subseteq L$, there exists a supervisor, which restricts the plant behavior to the desired behavior by dynamically disallowing some of the controllable events while never preventing any of the uncontrollable events from occurring, if and only if $K$ is controllable with respect to $L$. Moreover, when $K$ is not controllable, a minimally restrictive supervisor is designed which restricts the plant behavior to the supremal controllable sublanguage of $K$, denoted $K^\uparrow$.

Algorithms for testing controllability of $K$ with respect to $L$ and those for computing $K^\uparrow$ have been developed when both $K$ and $L$ are regular languages so that they are representable as finite state machine models [25, 1, 12, 10]. Recently Petri net (PN) models have received attention as an alternative model for investigating discrete event control theory (refer to [9] for a survey). PNs have a more descriptive power than finite state machines in the sense that the set of PN languages is a superset of regular languages and they allow a more concise model description.

In this paper we examine the controllability issues in the setting of deterministic PN languages [19]. Several interesting results already exist in literature. For example, it has been shown in [24] that the controllability of languages generated by PNs is in general undecidable; however, it is decidable in a restricted setting when the specification language is generated by a deterministic Petri net [5]. (This result extends the earlier results reported in [23] for the case of free-labeled nets, and in [22] for the case of deterministic nets.) Furthermore, it has been shown in [4] that the supremal controllable sublanguage of a PN language with respect to another PN language need not be a PN language.

In this paper we present an algorithm for computation of a minimally restrictive supervisor when the plant behavior is a deterministic PN language and the desired behavior is a regular language. This extends the prior results which apply to the case when plant and desired behavior are both regular languages. The algorithm due to Giua and DiCesare [3] which applies to the plant and the desired behaviors given as conservative PNs also belongs to this latter case since languages of conservative nets are regular. Our algorithm is based on a few results, also of independent interest, which we establish first. Specifically, we show that the problem of determining the controllability of a deterministic PN language $K$ with respect to another deterministic PN language $L$ is equivalent to determining the reachability of certain set of "bad" markings in a synchronized Petri net exhibiting the language $K \cap L$. A similar result is reported without proof in [6] (refer to Remark 2). $K$ is controllable with respect to $L$ if and only if these markings are unreachable in the synchronized net.

The set of bad markings is determined without having to construct any reachability tree of the synchronized net. Thus if $K$ is not controllable with respect to $L$, then a forbidden marking control that avoids reaching the bad markings can be used to achieve the supremal controllable sublanguage $K^\uparrow$ as the controlled plant behavior. Forbidden state avoidance control has been considered for several classes of Petri nets including marked graphs [7, 8] and nets without uncontrolled cycles [16, 17]. Thus these prior techniques can be applied for minimally restrictive control under the appropriate restrictions imposed in those works.

The topic of converting a language control problem into a state control problem has already been addressed some in the literature. Giua and DiCesare consider language control
through state avoidance of the synchronous composition of nets representing plant language $L$ and desired language $K$ [3]. They show that such a state control is possible when the synchronous composition is a conservative net. They do not provide a direct method for determining the set of markings to avoid, but rather suggest constructing a reachability tree to identify the set. As mentioned above, our work extends the above work. In [16] Li and Wonham associate a predicate of a suitably “refined” plant with the given language specification and show that the controller achieving the weakest controllable predicate [13] also achieves the supremal controllable language. They show that such a minimally restrictive control can be effectively computed by solving a linear integer programming problem when the plant is modeled as a vector discrete event system (or equivalently, a Petri net) satisfying the condition of loop-freedom and the desired predicate possesses a linear form [17]. Thus our work presented here, and also the work of Li and Wonham, extends the computation of a minimally restrictive controller from the setting of regular languages to more general but different settings.

2 Notation and Preliminaries

We use Petri nets to model a discrete event system. Letting $G_A$ denote a Petri net, it is defined to be the 5-tuple:

$$G_A := (\mathcal{P}_A, \mathcal{T}_A, \mathcal{E}_A, m_A^0, \alpha_A);$$

where $\mathcal{P}_A$ denotes the finite set of places of $G_A$; $\mathcal{T}_A$ is the finite set of transitions of $G_A$; $\mathcal{E}_A \subseteq \{\mathcal{P}_A \times \mathcal{T}_A\} \cup \{\mathcal{T}_A \times \mathcal{P}_A\}$ denotes the finite set of directed arcs connecting places and transitions of $G_A$; $m_A^0 \in \mathcal{N}^{\mathcal{P}_A}$ (where $\mathcal{N}$ is the set of natural numbers) denotes the initial marking\(^1\) of $G_A$; and $\alpha_A \in \Sigma^{\mathcal{T}_A}$ is the labeling function that associates with each transition $t \in \mathcal{T}_A$ an event $\alpha_A(t)$ belonging to the finite event set $\Sigma$. Given a pair of markings $m, m' \in \mathcal{N}^{\mathcal{P}_A}$; we say that $m$ covers $m'$ (equivalently, $m'$ is covered by $m$) if $m'(p) \leq m(p)$ for each $p \in \mathcal{P}_A$. Given a finite set $A$, we use the notation $A^*$ to denote the set of all finite sequences of elements in $A$ including the zero length sequence $\varepsilon$. The labeling function $\alpha_A$ is extended to $\alpha_A : \mathcal{T}_A^* \rightarrow \Sigma^*$ in a natural way.

For each $t \in \mathcal{T}_A$ we use the notation $t^{(p)}$ and $^{(p)}t$ to denote the set of “output” and “input” places of $t$ respectively, i.e.,

$$t^{(p)} := \{p \in \mathcal{P}_A \mid (t, p) \in \mathcal{E}_A\}; \quad ^{(p)}t := \{p \in \mathcal{P}_A \mid (p, t) \in \mathcal{E}_A\}.$$

Similarly, for each $p \in \mathcal{P}$, we use $p^{(T)}$, $(p)^T$ to denote the sets of transitions for which $p$ is an “input”, “output” place respectively, i.e.,

$$p^{(T)} := \{t \in \mathcal{T}_A \mid (t, p) \in \mathcal{E}_A\}; \quad ^{(T)}p := \{t \in \mathcal{T}_A \mid (t, p) \in \mathcal{E}_A\}.$$

\(^1\)The set $\mathcal{N}^{\mathcal{P}_A} = \{m \mid m : \mathcal{P}_A \rightarrow \mathcal{N}\}$ denotes the collection of all markings of $G_A$. 

3
A transition \( t \in T_A \) is said to be enabled if for all \( p \in \langle t \rangle, m(p) \geq 1 \). An enabled transition \( t \) may fire resulting in a new marking \( m' \in N^{P_A} \) given by the firing rule:

\[
m'(p) := m(p) - |p^{(T)} \cap \{t\}| + |p^{(T)} \cap \{t\}|
\]

Given \( m \in N^{P_A} \) and \( \tau \in T_A^* \) we use the notation \( m[\tau] \) to denote the marking reached by firing the transition sequence \( \tau \) starting at the marking \( m \) provided \( \tau \) is fireable in \( m \), else it is undefined.

Letting \( L(G_A) \) denote the language generated by \( G_A \), it is defined to be:

\[
L(G_A) := \{ s \in \Sigma^* | \exists \tau \in T_A^* \text{ s.t. } \alpha_A(\tau) = s \text{ and } m_A^0[\tau] \text{ is defined} \}
\]

\( G_A \) is called a deterministic PN, and its generated language \( L(G_A) \) is called a deterministic PN language if for each \( s \in L(G_A) \), \(|\alpha_A^{-1}(s)| = 1 \). It is clear that \( L(G_A) \) is prefix closed \([21]\).

In \([20]\), the language generated is referred to as a P-type language. For the definition of other types of languages of Petri nets refer to \([20]\), and for relations among various languages of Petri nets refer to \([15]\) and references therein.

Next we define synchronous composition of two Petri nets \( G_P \) and \( G_S \), which is useful in our analysis.

**Definition 1** The synchronous composition of Petri nets \( G_P := (P_P, T_P, E_P, m^0_P, \alpha_P) \) and \( G_S := (P_S, T_S, E_S, m^0_S, \alpha_S) \) is another Petri net \( G_P \| G_S := G := (P, T, E, m^0, \alpha) \), where

- \( P := P_P \cup P_S \),
- \( T := \{ (t_p, t_s) \in T_P \times T_S | \alpha_P(t_p) = \alpha_S(t_s) \} \);
- \( \alpha((t_p, t_s)) := \alpha_P(t_p) = \alpha_S(t_s) \),
- \( E := \{ (p, (t_p, t_s)) \in P \times T | (p, t_p) \in E_P \text{ or } (p, t_s) \in E_S \} \cup \{ ((t_p, t_s), p) \in T \times P | (t_p, p) \in E_P \text{ or } (t_s, p) \in E_S \} \),
- \( m^0(p) := \begin{cases} m^0_P(p) & \text{if } p \in P_P \\ m^0_S(p) & \text{if } p \in P_S \end{cases} \)

The set of places in the synchronized net equals the union of those in the individual nets. The synchronized net replaces a pair of transitions with the same label but in separate nets with a single transition in the new net. This single transition has the “input” places and “output” places as respective union of those from the previous transition pair. Note that there may exist several transitions in each net with the same label, in which case, there exists one transition in the synchronized net for each transition pair combination. It is straightforward to show that the language of the synchronized net \( G \) satisfies \( L(G) = L(G_P) \cap L(G_S) \) \([20]\). Also, \( G \) is deterministic whenever \( G_P \) and \( G_S \) are deterministic.

For supervisory control the event set \( \Sigma \) is partitioned into \( \Sigma = \Sigma_u \cup (\Sigma - \Sigma_u) \), the sets of uncontrollable and controllable events. Given a discrete event plant with generated behavior \( L \subseteq \Sigma^* \), and a prefix closed language \( K \subseteq L \) representing the desired behavior, the supervisory control problem is to synthesize a controller, also called a supervisor, which dynamically
disables some of the controllable events while never preventing any uncontrollable event from occurring, so that the controlled plant behavior equals $K$. Since a supervisor cannot prevent an uncontrollable event, such a control task is feasible if and only if $K$ is controlable with respect to $L$, i.e., if and only if the following invariance relation holds:

$$K \Sigma_u \cap L \subseteq K.$$  

We say that a string $s \in K$ is an uncontrollable string if there exists an uncontrollable event $\sigma_u \in \Sigma_u$ such that $s \sigma_u \in (L - K)$. It is clear that $K$ is controllable if and only if it contains no uncontrollable strings.

## 3 Language Control via Forbidden Marking

In this section we prove that if the specification of the plant behavior and that of the desired behavior are both given as Petri net languages, then the problem of determining controllability of the desired behavior with respect to the plant behavior is equivalent to that of determining the reachability of a certain set of markings from the initial marking in the synchronized net. We also show that the problem of minimally restrictive supervision is reducible to that of a forbidden marking problem in the synchronized net. These results are used in the following section to obtain an algorithm for computing a minimally restrictive control.

As before, let $L \subseteq \Sigma^*$ be a deterministic PN language representing the generated behavior of a discrete event plant, and $K \subseteq L$ be another prefix closed deterministic PN language representing the desired behavior of the controlled plant. Let $G_L := (\mathcal{P}_L, \mathcal{T}_L, \mathcal{E}_L, m^0_L, \alpha_L)$ and $G_K := (\mathcal{P}_K, \mathcal{T}_K, \mathcal{E}_K, m^0_K, \alpha_K)$ be two deterministic Petri nets such that $L(G_L) = L$ and $L(G_K) = K$, respectively. We assume without loss of generality that the domain of $\alpha_L$ equals the domain of $\alpha_K$. Note that this can be accomplished even when the events in language $K$ are a strict subset of the events in language $L$, since $G_K$ can contain transitions which will never be enabled.

Consider the Petri net $G_L || G_K := G := (\mathcal{P}, \mathcal{T}, \mathcal{E}, m^0, \alpha)$ obtained by synchronous composition of $G_L$ and $G_K$. Then by construction of $G$, we obtain $L(G) = L(G_L) \cap L(G_K) = L \cap K = K$, where the last equality follows from the fact that $K \subseteq L$. We partition the transition set $\mathcal{T} = \mathcal{T}_u \cup (\mathcal{T} - \mathcal{T}_u)$ into the sets of uncontrollable transitions and the set of controllable transitions, where $\mathcal{T}_u := \{t \in \mathcal{T} \mid \alpha(t) \in \Sigma_u\}$.

Define the following set of “bad” markings of the synchronized net $G$:

$$M_b := \{m \in \mathcal{N}^\mathcal{P} \mid \exists t = (t_L, t_K) \in \mathcal{T}_u \text{ s.t. } m[t] \text{ is undefined and } (m|_{\mathcal{P}_L})[t_L] \text{ is defined}\},$$

where $m|_{\mathcal{P}_L}$ denotes the projection of the marking $m$ of $G$ to the marking of $G_L$. Thus $m \in M_b$ if and only if there exists an uncontrollable event that is not enabled in marking $m$ of $G$, whereas it is enabled in the corresponding marking $m|_{\mathcal{P}_L}$ of $G_L$. Thus $M_b$ denotes those markings of $G$ in which an uncontrollable event is allowed in the plant behavior and disallowed in the desired behavior.
Remark 1 Note that if a marking \( m_1 = (m_1^L, m_1^K) \in \mathcal{N}^P_L \times \mathcal{N}^P_K \) is such that \( m_1 \in M_b \), then any marking \( m_2 = (m_2^L, m_2^K) \in \mathcal{N}^P_L \times \mathcal{N}^P_K \) such that \( m_2^L \) covers \( m_1^L \) and \( m_2^K \) is covered by \( m_1^K \) is also in \( M_b \). We exploit this property of the set of bad markings to develop an algorithm for computation of a minimally restrictive control in Section 4.

Theorem 1 Let \( K \subseteq L \subseteq \Sigma^* \) be prefix closed deterministic Petri net languages, and let \( M_b \) be as defined above. Then \( K \) is controllable with respect to \( L \) if and only if no marking in \( M_b \) is reachable from the initial marking \( m^0 \) of the synchronized net \( G = G_L \| G_K \).

Proof: We prove the contrapositive of Theorem 1, i.e., we prove that \( K \) is not controllable with respect to \( L \) if and only if a marking in \( M_b \) is reachable from \( m^0 \).

First assume that there exists a marking \( m \in M_b \) such that \( m \) is reachable from \( m^0 \), i.e., there exists a transition sequence \( \tau \in \mathcal{T}^* \) such that \( m^0[\tau>] \in M_b \). Then we need to show that \( K \) is not controllable with respect to \( L \), i.e., we need to show that there exists a string \( s \in K \) and an uncontrollable event \( \sigma_u \in \Sigma_u \) such that \( s \sigma_u \in (L - K) \). We claim that \( \alpha(\tau) \) is such a string. Since \( m^0[\tau>] \) is defined, it is clear that \( \alpha(\tau) \in L(G) = K \). Also, since \( m^0[\tau>] \in M_b \), there exists a transition \( t = (t_L, t_K) \in \mathcal{T}_u \subseteq \mathcal{T}_L \times \mathcal{T}_K \) such that \( (m^0[\tau>])t = m^0[\tau t>] \) is not defined and \( (m^0[\tau>]|_{\mathcal{P}_L} t> \) is defined. Since the nets are deterministic, this implies that \( \alpha(\tau) \notin K \) and \( \alpha(\tau t) \in L \). Thus \( \alpha(\tau) \) is an uncontrollable string.

Next we prove that if \( K \) is not controllable with respect to \( L \), then there exists a transition sequence \( \tau \in \mathcal{T}^* \) such that \( m^0[\tau>] \in M_b \). Since \( K \) is not controllable with respect to \( L \), there exists a string \( s \in K \) and an uncontrollable event \( \sigma_u \in \Sigma_u \) such that \( s \sigma_u \in (L - K) \). Since \( s \in K = L(G) \), \( \alpha^{-1}(s) \) is defined. We claim that \( m^0[\alpha^{-1}(s>] \in M_b \). Suppose for contradiction that \( m^0[\alpha^{-1}(s>] \notin M_b \). Then from the definition of \( M_b \), for the string \( s \in K \), there exists no \( t \in \mathcal{T}_u \) such that \( s \sigma(t_u) \in (L - K) \). This contradicts the earlier statement that the string \( s \) is an uncontrollable string.

Remark 2 A result similar to that of Theorem 1 is stated without proof in [6], where the set of bad markings is defined slightly differently:

\[
\{ m \in \mathcal{N}^P \mid \exists t = (t_L, t_K) \in \mathcal{T}_u \text{ s.t. } (m|_{\mathcal{P}_L})t_L> \text{ is defined and } (m|_{\mathcal{P}_K})t_K> \text{ is undefined} \}.
\]

However, the result is only partially correct because the assumption of determinism is not imposed, which is needed for the necessity part of the result to hold. It is clear that this set of markings is identical to \( M_b \). The set of bad markings \( M_b \) can be easily identified by inspecting the input places of each uncontrollable transition \( t \in \mathcal{T}_u \) in the synchronized net \( G \). However, it is not clear whether their reachability is decidable since in general \( M_b \) is an infinite set, and the reachability of only a finite set of markings is known to be decidable [18]. But using the result of Sreenivas that controllability is decidable in the setting of deterministic PN languages (or its extension reported in [5]) we can conclude that the reachability of \( M_b \) is decidable. Finally, since \( K \) is controllable if and only if its prefix closure is controllable, Theorem 1 can be easily extended to the case when \( K \) is non-prefix closed, but its prefix closure is a deterministic PN language.
Next we show that the problem of minimally restrictive supervision can be reduced to a forbidden marking problem. Since $K, L$ are both assumed to be prefix closed, it follows from [12, Corollary 3.3] that $K$ is controllable with respect to $L$ if and only if $K\Sigma_u \cap L \subseteq K$. Using this result and Theorem 1 we obtain the result of the next theorem. We first define the following set of markings of $G$:

$$M'_b := \{m \in \mathcal{N}^P \mid \exists \tau \in T_u^* \text{ s.t. } m[\tau] \in M_b\}.$$ 

$M'_b$ is the set of markings from which the system can uncontrollably reach the bad marking set $M_b$. The result of the following lemma is straightforward:

**Lemma 1** Let $K \subseteq L \subseteq \Sigma^*$ be prefix closed deterministic Petri net languages and $G_L$ and $G_K$ be as defined above. Then $K$ is controllable with respect to $L$ if and only if no marking in the set $M'_b$ is reachable from $m^0$, the initial marking of $G$.

The following corollary is immediate from Lemma 1:

**Corollary 1** Let $K \subseteq L \subseteq \Sigma^*$ be prefix closed deterministic Petri net languages and $G_L$ and $G_K$ be as defined above. Then $s \in K$ is uncontrollable if and only if $m^0[\alpha^{-1}(s)] \in M'_b$.

Corollary 1 characterizes the set of uncontrollable strings of $K$. We prove in the next theorem that if all those strings in $K$, for which execution of any prefix leads to a marking in $M'_b$, are deleted from $K$, then the remaining language equals $K^\uparrow$.

**Theorem 2** Let $K \subseteq L \subseteq \Sigma^*$ be prefix closed deterministic PN languages and let $M'_b$ be as defined above. Then

$$K^\uparrow = K - B\Sigma^*,$$

where $B := \{s \in K \mid m^0[\alpha^{-1}(s)] \in M'_b\}$.

**Proof:** For notational simplicity, define $K - B\Sigma^* := H$. Then we need to prove that $K^\uparrow = H$. Note that $K - B\Sigma^*$ is prefix closed since $K$ is prefix closed.

We first prove that $H \subseteq K^\uparrow$. Since $K^\uparrow$ is the supremal controllable sublanguage of $K$, it suffices to show that $H$ is controllable, i.e., $HS \cap L \subseteq H$ (as $H, L$ are prefix closed). Pick any string $s \in H$, then we need to show that there does not exist a $\sigma_u \in \Sigma_u$ such that $s\sigma_u \in (L - H)$. Suppose for contradiction that there exists a $\sigma_u$ such that $s\sigma_u \in (L - H)$. Since $H \subseteq K \subseteq L$, $L - H = (L - K) \cup (K - H)$. Thus either $s\sigma_u \in (L - K)$ or $s\sigma_u \in (K - H)$. If $s\sigma_u \in K - H$, then by definition of $H$ and Corollary 1, there exists $u \in \Sigma_u^*$ such that $s\sigma_u u \in L - K$. Thus in either case there exists a sequence of uncontrollable events which when appended to $s$ results in a string of $L - K$. This shows that in either case $m^0[\alpha^{-1}(s)] \in M'_b$, which contradicts that $s \in H$.

Next we prove that $K^\uparrow \subseteq H$. Since both $K^\uparrow, H \subseteq K$, it suffices to show that $(K - H) \subseteq (K - K^\uparrow)$. Pick $s \in (K - H)$, then there exists a prefix $s'$ of $s$ such that $m^0[\alpha^{-1}(s')] \in M'_b$. It then follows from Corollary 1 that $s$ is uncontrollable. Hence $s' \in (K - K^\uparrow)$, which implies that $s \in (K - K^\uparrow)$.

\[\Box\]
Remark 3 If we let \( M_b \) denote the set of forbidden markings, then \( M_b^\ast \) is the set of weakly forbidden markings as defined in \([7]\). From discussions in \([7]\), it follows that a minimally restrictive control that avoids the markings in the set \( M_b \) also avoids the markings in the set \( M_b^\ast \). This together with Theorem 2 implies that such a control achieves the supremal controllable sublanguage \( K^\dagger \). Thus we have reduced the problem of constructing the generator for \( K^\dagger \) to a forbidden marking problem, the markings of which can be easily identified. Consequently known forbidden marking algorithms such as those discussed in the introduction can be applied under the appropriate restrictions of those algorithms.

4 Algorithm for Minimally Restrictive Control

In this section we provide an algorithm for computing a minimally restrictive control under the additional condition that \( K \) can be generated by a safe Petri net, i.e., when it is regular. Recall that a Petri net is said to be safe if the number of tokens in any place of any of its reachable marking does not exceed one. Consequently, the set of reachable markings for a safe net is a finite set. Without loss of any generality this net can be chosen to be a deterministic net.

Note that each marking in \( M_b \) is of the form \( m_b = (m_b^L, m_b^K) \in \mathcal{N}^{P_L} \times \mathcal{N}^{P_K} \), where \( m_b^L \) is the marking of places in \( P_L \), and \( m_b^K \) is the marking of the places in \( P_K \). Due to the safeness of \( G_K \), \( m_b^K \) can only take a finitely many values. On the other hand, \( m_b^L \) can take infinitely many values, but it suffices to consider the “minimal” elements of this set. Define the set \( (M_b^L)^{\min} \) as follows:

\[
(M_b^L)^{\min} := \{ m_b^L \mid \exists t \in \mathcal{T}_u \text{ s.t. } m_b^L(p) = 1, \forall p \in P_L \cap (p)t; \text{ and } m_b^L(p) = 0, \text{ otherwise} \}.
\]

In other words, \( m_b^L \in (M_b^L)^{\min} \) if there exists an uncontrollable transition which is enabled under the marking of \( m_b^L \), and it is a minimal such marking (any other marking under which the corresponding uncontrollable transition is enabled covers this marking). It then follows from the definition of \( M_b \) that the set \( (M_b^L)^{\min} \) is the set of minimal elements of the set of markings of \( G_L \) that correspond to bad markings of \( M_b \). It also follows that the cardinality of \( (M_b^L)^{\min} \) is less than or equal to the number of uncontrollable transitions, \( \mathcal{T}_u \), so \( (M_b^L)^{\min} \) is finite. Using these observations, we next show that by augmenting the synchronized net \( G \) with certain “complementary” places, it is possible to determine the reachability of a marking in \( M_b \) by determining the coverability of a certain finite set of markings in the augmented net.

In the following definition the notation \( \bar{p} \) is used to denote a complementary place for a place \( p \) of the synchronized net \( G \). As the name suggests, these complementary places are marked if and only if the corresponding places are unmarked. A complementary place is added for each place in \( P_K \). A complementary place \( \bar{p} \) is made an output place (respectively, an input place) of a transition \( t \in \mathcal{T} \) whenever the associated place \( p \in P_K \) is an input place (respectively, an output place) of the transition \( t \). Formally,
Definition 2 The augmented net $G^A$, for the the synchronized net $G := (P, T, E, m^0, \alpha)$, is defined to be: $G^A := (P \cup \bar{P}_K, T, E \cup \bar{E}, (m^A)^0, \alpha)$, where

- $\bar{P}_K := \{ \bar{p} \notin P \mid p \in P_K \}$,
- $\bar{E} := \{(t, \bar{p}) \in T \times \bar{P}_K \mid p \in ({}^p t - {}^p t) \} \cup \{(\bar{p}, t) \in \bar{P}_K \times T \mid p \in t^p - {}^p t \}$,
- $(m^A)^0(p) := m^0(p), \forall p \in P$; \quad $(m^A)^0(\bar{p}) := -m^0(p), \forall \bar{p} \in \bar{P}_K$.

The result of the following lemma can be obtained using induction on the length of firing sequences in a straightforward manner:

Lemma 2 Consider $G := G_L \mid G_K$, where $G_K$ is a safe PN, and $G^A$ as defined above. Then for any $\tau \in T^*$:

1. $m^0[\tau]$ is defined in $G$ if and only if $(m^A)^0[\tau]$ is defined in $G^A$, and
2. If $m = m^0[\tau]$ and $m^A = (m^A)^0[\tau]$, then $m^A(p) = m(p)$ for $p \in P$ and $m^A(\bar{p}) = -m(p)$ for $\bar{p} \in \bar{P}_K$.

Using the result of Lemma 2 we give an algorithmic test for controllability in the next theorem, which is then used to compute a minimally restrictive control. We first define a certain set of markings of the augmented net $G^A$. Note that a marking of the augmented net is of the type $m = (m^L, m^K, \hat{m}^K)$, where $m^L$ is the marking of places in $P_L$, $m^K$ is the marking of places in $P_K$, and $\hat{m}^K$ is the marking of the complementary places $\bar{P}_K$. Define the following set of markings of the augmented net:

$$M^A_b := \{(m^L_b, m^K_b, \hat{m}^K_b) \mid (m^L_b, m^K_b) \in M_b \text{ and } m^L_b \in (M^L_b)^{\text{min}} \}.$$ 

Thus $M^A_b$ consists of markings of the augmented net which correspond to those markings in $M_b$ that are minimal for the places in $P$. Note that $M^A_b$ is finite since $G_K$ is a safe PN and since $(M^L_b)^{\text{min}}$ is finite.

Theorem 3 Consider $G := G_L \mid G_K$ and $G^A$ as defined above, where $G_K$ is safe. Then a marking in $M_b$ is reachable in $G$ if and only if a marking in $M^A_b$ is coverable in $G^A$.

Proof: First suppose $m_b = (m^L_b, m^K_b) \in M_b$ is reachable in $G$. Then it follows from Lemma 2 that $m^A_b = (m^L_b, m^K_b, \hat{m}^K_b)$ is reachable in $G^A$. Since $m_b \in M_b$, it follows that $m^L_b$ covers some minimal marking in $(M^L_b)^{\text{min}}$. So clearly $m^A_b$ covers a marking belonging to $M^A_b$.

Conversely, suppose a marking in $M^A_b$ is coverable in $G^A$. This implies that there exists a marking $m^A_b = (m^L_b, m^K_b, \hat{m}^K_b)$ that covers a certain marking $(m^{L'}_b, m^{K'}_b, \hat{m}^{K'}_b) \in M^A_b$ and it is reachable in $G^A$. Then it follows from Lemma 2 that $m_b = (m^L_b, m^K_b)$ is reachable in $G$. We need to show that $m_b \in M_b$. Due to the complementary nature of $m^K_b$ and $\hat{m}^K_b$, $(m^K_b, \hat{m}^K_b)$ can only cover itself. Hence $m^K_b = m^{K'}_b$ and $\hat{m}^K_b = \hat{m}^{K'}_b$. Since $(m^{L'}_b, m^{K'}_b, \hat{m}^{K'}_b) \in M^A_b$, by definition we have $(m^{L'}_b, m^{K'}_b) \in M_b$. Since $m^{K'}_b = m^K_b$, this implies $(m^{L'}_b, m^K_b) \in M_b$. Finally, since $m^L_b$ covers $m^L_b$, it follows from discussions of Remark 1 that $(m^L_b, m^K_b) \in M_b$, as desired.

The following on-line scheme can be used for minimally restrictive supervision: Enable a controllable transition if and only if the resulting marking is such that a marking in $M^A_b$ is not uncontrollably coverable from it.
Algorithm 1 Consider prefix closed $K$ and $L$, where $K$ is regular and $L$ is a deterministic PN language, and the augmented net $G^A$.

1. **Initialization step:**
   Set the current marking $m$ to be $(m^A)^0$.

2. **Control step:**
   For each $t \in T - T_u$ such that $m[t]$ is defined in $G^A$, disable $t$ if and only if a marking in $M_b^A$ is coverable in $G^A$ by a sequence of uncontrollable transitions from the marking $m[t]$. Update the current marking $m$ upon observing the occurrence of an event (that is not disabled), and repeat the control step.

It follows from Theorem 3 that control of Algorithm 1 prevents the reachability of markings $M^*_b$ in the synchronized net $G$. Hence it follows from Theorem 2 that the controlled plant behavior equals $K^\dagger$.

**Remark 4** The computation of minimally restrictive supervisor is *on-line* in nature and requires the testing of coverability of the finite set of markings in $M_b^A$, the computational complexity of which is in general exponential in the number of places of the associated net [2]. However, note that the number of states in the plant Petri net can be infinite, in which case an automata theoretic approach for the computation of the supervisor is not possible. In the special case when the number of states in the plant Petri net is finite, it can be exponential in the number of places, and an automata theoretic approach will have a comparable computational complexity. Finally, using a recent result of Kumar-Garg [11] that a minimally restrictive control for avoidance of a *right-closed* set of markings is effectively computable, it can be concluded that an *off-line* computation of the minimally restrictive supervisor in the present setting is also possible, although the exact computational complexity of this off-line computation remains to be determined.

5 **Conclusion**

In this paper we have obtained an algorithm for computing a minimally restrictive control when the plant behavior is a deterministic PN language and the desired behavior is regular language extending the earlier results of supervisory control theory. Our algorithm requires prevention of coverability of a finite set of markings $M_b^A$ which can easily be identified by inspecting the net obtained by the synchronous composition of the plant and the desired behavior nets. The set $M_b^A$ of markings to avoid includes only binary markings, so any control policy to prevent them from being covered is equivalent to enforcing a “forbidden condition” [9]. Such forbidden condition control has been proposed for a variety of net classes under various conditions [9], and in some cases these can be exploited directly to prevent the coverability of markings in $M_b^A$. 


References


