Sequential Synthesis of Supervisory Policies for Discrete-Event Systems Modeled by Petri Nets

A. Raman and R. S. Sreenivas

Abstract—It is often of interest to synthesize a supervisory policy for enforcing complex properties on the behaviour of a Discrete-Event System (DES). One way of doing this is by decomposing complex properties into simpler objectives and then synthesizing supervisors for those simpler objectives in a sequential manner. This approach is particularly convenient if the supervised-system can be represented using the same modeling framework at each stage of this sequential process. An additional desirable feature could be that the supervisory policy remain the same even if the initial-state of the DES were to change.

In this paper, we consider Petri Net (PN) models of Discrete-Event Systems (DES) under a supervisory policy that enforces a desired-property $B$. We prove that the supervised-system can be modeled as a PN if and only if the supervisory policy is a marking-monotone-$B$-enforcing supervisory policy (MM-$B$ESP) over reachable markings. In the second half of the paper we describe a software tool for the synthesis of MM-ESPs, where the desired-property $B$ is the PN-property of liveness, for arbitrary Petri Nets. We end the paper with an example that illustrates both the contributions.

I. INTRODUCTION

A Discrete Event System (DES) is a discrete-state system, where the discrete-state changes at discrete-time instants due to the occurrence of events. Suppose we have a directed graph representing a DES in which each node represents a state of the system, and each edge represents an event that takes the system from one state to another. Suppose some of the events in the system are controllable, in the sense that they can be prevented from occurring by a supervisor. The supervisory policy specifies which events (edges) to disable (resp. remove) at which states (resp. nodes) such that the DES (residual graph) satisfies a desired property $B$. The simplest way of doing this is to use a procedure that tries all possible combinations of edges that can be removed. However, this approach is inapplicable when the graph is infinite in size. Even if we have a finite-model of the infinite-state DES, there is the additional requirement for the existence of a procedure that synthesizes the supervisory policy for this DES. Although well established methods exist for synthesizing policies that enforce certain properties (like liveness, safety, boundedness etc.), it is likely that there do not not exist systematic procedures for the synthesis of policies that enforce complex objectives (like the combination of several objectives). One of the ways of accomplishing this is by decomposing the complex objective into simpler objectives and then synthesizing policies for enforcing these simpler objectives (cf. Figure 1).

The issue with this approach is that while there was a model, within a modeling-paradigm, of the original infinite-state DES, there may not exist a model of the supervised system within the same modeling-paradigm. For instance, Example 3.1 of [1] presents a plant DES that is a Petri Net (PN), where the desired behavior $B$ corresponds to the requirement to be non-blocking. Giua and DiCesare have shown that there is no PN that can model the resulting supervised-system. In the context of Figure 1 of this paper, this result shows that while the plant for Supervisor 1 is a PN, the plant for Supervisor 2 cannot be PN in any sequential attempt to the synthesis of supervisory policies. The identification of a necessary and sufficient condition for a DES modeling paradigm (different from Finite State Automata) where the supervised-DES can also be represented using the same modeling paradigm, plays a critical role in the sequential synthesis of supervisory policies for DES. Thus, there are two aspects to this problem— (i) the algebraic framework under which the supervisor construction is carried out; and, (ii) the model. We discuss each of these points in the remainder of the section.

Modular problem specification and supervisor construction for DES was first discussed by Ramadge and Wonham in [2], [3]. They modeled the behavior of DES as a prefix closed language $L$ over the set of event alphabet $\Sigma$, where each $u \in L$ is a possible event sample path. The behaviour of the language $L$ is modeled by a generator $G$ which is an automaton $(\Sigma, Q, \delta, q_0)$. Here $Q$ represents the set of (possibly infinite) states, $\delta : \Sigma \times Q \rightarrow Q$ is the transition function and $q_0$ is the initial state. Control is modeled by partitioning the event space into controllable and uncontrollable events: $\Sigma = \Sigma_c \cup \Sigma_u$. For multiple objective controller synthesis, if each objective is specified in terms of a controlled language $K_i$, then the overall desired behaviour

![Fig. 1. Sequential Synthesis of Supervisory Policies.](image-url)
is specified by the controlled language $\cap_i K_i$. There are two key points—controllability of the desired language and compatibility between multiple objectives. For prefix closed languages, if $K_1$ and $K_2$ are controllable, then $K_1 \cap K_2$ is also controllable (see Section IV in [3] for the definition of controllability). Compatibility between multiple objectives is formalized by the concept of nonconflicting languages. Two controlled languages $K_1, K_2 \in \Sigma^*$, are said to nonconflicting if $pr(K_1 \cap K_2) = pr(K_1) \cap pr(K_2)$ where $pr(\bullet)$ denotes the prefix of the string argument. That is, whenever the two languages $K_1$ and $K_2$ share a prefix, they also share a word containing this prefix.

Modeling of a supervised Petri net by another Petri net was first discussed in [4], where the authors gave an algorithm for constructing the PN model of the supervised system. They proved that such a construction can always be carried out for conservative PNs (cf. Theorem 4.1 in [4]). Reference [1] proved by a counter-example that not all supervised Petri nets can be modeled by a PN. Using the work in the aforementioned papers as a starting point, in this paper we first formally define the concept of composition of a PN model and a supervisory policy. We extend the algorithm in [4] to unbounded PNs, and prove that a composition of a PN model and a supervisory policy $\mathcal{P}$ that enforces a property $B$ exists if and only if the supervisory policy is marking-monotone $B$-enforcing supervisory policy (MM-BESP) over the reachable markings. That is, (i) if $\mathcal{P}$ permits a transition to fire at a smaller marking, then it permits the transition at all reachable larger markings as well, and (ii) $\mathcal{P}$ enforces property $B$ at all reachable larger markings. We also generalize the concept of marking-monotonicity from the set of reachable markings to all markings in the marking space and prove the necessary and sufficient conditions for the existence of an MM-BESP. By definition, if $\mathcal{P}$ is an MM-BESP for an initial marking, then it is an MM-BESP for all larger initial markings as well. Since the policy does not change for a larger initial marking, the composition of the PN and the policy also stays the same for a larger initial marking. This is an important property to have while designing a system as analysis for various initial markings become easy. This behaviour is different compared to other well studied properties like safety, boundedness, reachability of a marking etc. For instance, if a PN is bounded under the supervision of a policy, then trimming the set of reachable markings will not result in loss of boundedness. As the last contribution of the paper, we combine marking monotonicity with liveness and describe a software tool for the synthesis of marking-monotone $\mathcal{L}$ESP for arbitrary PNs.

The paper is organized as follows. In Section II we formally discuss some preliminaries of PNs and introduce notations and definitions that we will use in the rest of the paper. Section III presents the main results of this paper. We describe the software tool for the synthesis of an MM-ESP in Section IV. Section V presents an example to illustrate the results. We conclude the paper with Section VI.
 markings under the supervision of $P$. A supervisory policy \(\mathcal{P}\) is \(\text{MM-}\)monotone if all transitions in $\mathbb{M}$ are right-closed, i.e., if for some $c\in\mathbb{M}$, $c\xrightarrow{\mathbb{M}}\mathbb{m}_t$. Therefore, a supervisory policy that enforces a right-closed transition $t$ at state $\mathbb{m}_t$ is also a \(\text{BESP}\) for $N(\mathbb{m}_t)$.

**III. MAIN RESULTS**

**Definition 1:** Let $\mathcal{P}$ be a \(\text{BESP}\) for $N(\mathbb{m}_0)$, $N(\mathbb{m}_0)$ and $N(\mathbb{m}_0, \mathcal{P})$ are said to be \(\text{B-composable}\) if there exists a Petri net $N_2$ such that $\mathcal{R}(N_2, \mathbb{m}_0) = \mathcal{R}(N_1, \mathbb{m}_0, \mathcal{P})$. We refer to $N_2(\mathbb{m}_0)$ as the \(\text{B-preserving composition of } N(\mathbb{m}_0)\) and $\mathcal{P}$.

**Theorem 1:** There exists an \(\text{MM-BESP}\) for $N(\mathbb{m}_0)$ if and only if there exists a subset $\hat{\mathcal{P}}(B, N)\subseteq\mathcal{D}(B, N)$ such that:

1. $\mathbb{m}_0 \in \hat{\mathcal{P}}(B, N)$.
2. $\hat{\mathcal{P}}(B, N)$ is right-closed.
3. A supervisory policy that enforces the set $\hat{\mathcal{P}}(B, N)$ is a \(\text{BESP}\).

**Proof:** (If) Suppose $\hat{\mathcal{P}}(B, N)$ is right-closed and $\mathbb{m}_0 \in \hat{\mathcal{P}}(B, N)$. Let $\{\mathbb{m}_i\}_{i=1}^l$ denote the minimal elements of $\hat{\mathcal{P}}(B, N)$. Consider a supervisory policy that enforces the set $\hat{\mathcal{P}}(B, N)$, that is:

Let $\mathbb{m}_0 \geq \mathbb{m}_i^*$ and consider a transition $t$ such that $\mathbb{m}_0 \xrightarrow{t} \mathbb{m}_i^*$ and $\mathbb{m}_0 \xrightarrow{t} \mathbb{m}_i^*$. Suppose $\mathcal{P}(\mathbb{m}_0, t) = 1$. Then $\mathbb{m}_i^* \geq \mathbb{m}_0 \geq \mathbb{m}_i^*$. Therefore, a supervisory policy that enforces a right-closed transition $t$ at state $\mathbb{m}_0$ is also a \(\text{BESP}\) for $N(\mathbb{m}_0)$.

Let $\hat{\mathcal{P}}(B, N) = \mathcal{D}(B, N)$, then $\hat{\mathcal{P}}(B, N)$ is right-closed by definition. Since there is a \(\text{MM-BESP}\) for $N(\mathbb{m}_0)$, it follows that $\mathbb{m}_0^* \in \hat{\mathcal{P}}(B, N)$. Let $\mathbb{m}_0^* \xrightarrow{t} \mathbb{m}_0^*$ for some $t_0 \in T_u$. Then we have that $\mathbb{m}_0^* \in \hat{\mathcal{P}}(B, N)$. If not, the supervisory policy will not be an \(\text{MM-BESP}\) for $\mathbb{m}_0^*$ (as $\hat{\mathcal{P}}(B, N) = D(M(\mathbb{m}_0))$). In fact, using the same argument, the \(\text{MM-BESP}\) will disable any controllable transition whose firing takes the PN outside $\hat{\mathcal{P}}(B, N)$. Therefore, the marking monotone policy that enforces the set $\hat{\mathcal{P}}(B, N)$ is a \(\text{BESP}\).
observation that it is the largest non-negative marking greater than \((\vec{m} - C \times 1_c)\) and that \(\vec{m}^i\) is the minimal element of \(\mathcal{D}_M(\mathcal{B}, N)\).

**Algorithm 1 CO\-MPOSE\((N_1, \mathcal{D}_M(\mathcal{B}, N_1))\)**

1. \(\Pi_2 = \Pi_1\)
2. \(T_2 = T_{1u}\)
3. \(\Gamma_2(p, t) = \Gamma_1(p, t)\) \(\forall t \in T_{1u}\)
4. \(\text{for } i \in \{1, \ldots |T_{1u}|\} \text{ do}
5. \(\text{for } j \in \{1, \ldots k\} \text{ do}
6. \(T_2 \leftarrow T_2 \cup \{t^j_i\}\)
7. \(\text{end for}\)
8. \(\Gamma(p, t^j_i) = (\max(0, \vec{m}^j_i - C \times 1_i))(p)\)
9. \(\Gamma(t^j_i, p) = (\max(0, \vec{m}^j_i - C \times 1_i) + C \times 1_i)(p)\)
10. \(\text{end for}\)
11. \(\text{end for}\)
12. \(\text{end for}\)

Algorithm 1 presents a procedure for evaluating a \(B\)-preserving composition. \(N_2 = (\Pi_2, T_2, \Gamma_2)\), of a PN \(N_1 = (\Pi_1, T_1, \Gamma_1)\) and an MM-BESP, \(P_M\), that enforces the set \(\mathcal{D}_M(\mathcal{B}, N_1)\). We need to construct \(N_2\) such that \(\mathcal{R}(N_2, m^0) = \mathcal{R}(N_1, m^0, P_M)\). A supervisory policy has no control over the uncontrollable transitions. Therefore, intuitively, the behaviour of the system in the uncontrollable space should be the same for \(N_1\) and \(N_2\) (Steps 2 and 3). Lemma 1 identifies \(k\) minimal elements of the set \(\mathcal{T}_M(\mathcal{B}, N_1, t)\) for a controllable transition \(t_i\) of \(N_1\). Using this observation, each controllable transition, \(t_i\), of \(N_1\) is replaced by \(k\)-many controllable transitions, \(\{t^j_i\}_{j=1}^k\) in \(N_2\). The input arc-weights of \(\{t^j_i\}_{j=1}^k\) correspond to these minimal elements of \(\mathcal{T}_M(\mathcal{B}, N_1, t_i)\). The output arc-weights of \(\{t^j_i\}_{j=1}^k\) correspond to the effect of firing \(t_i\) steps. Steps 4 to 12 accomplish these tasks. \(T_{1c}\) and \(T_{1u}\) denote the set of controllable and uncontrollable transitions in \(N_1\) respectively.

MM-BESPs are a generalized version of MM-BESPs over-reachable-markings. While MM-BESPs consider the marking monotonicity over all markings, MM-BESPs over reachability only consider the markings that are reachable from the initial marking (that is, ignoring the markings that are not reachable from the initial marking). It follows that the existence of an MM-BESP implies the existence of an MM-BESPs over reachability. Lemma 1 and Algorithm 1 can be easily extended for MM-BESPs over reachable markings by constraining the analysis to reachability markings. We do not explicate the details in the interest of space.

**Theorem 2:** For an arbitrary Petri Net \(N_1\): \((\exists \text{ a } B\text{-preserving composition of } N_1(m^0) \text{ and } P) \Leftrightarrow (P \text{ is an MM-BESP over reachable markings}).

**Proof:** \((\Rightarrow)\) Suppose there exists a \(B\)-preserving composition of \(N_1(m^0)\) and \(P\). Then \(\mathcal{R}(N_2, m^0) = \mathcal{R}(N_1, m^0, P)\). Since we have to consider the unsupervised reachability graph of \(N_2\), without loss of generality in the context of the proof, we assume all transitions in \(N_2\) are uncontrollable. The condition \(\mathcal{R}(N_2, m^0) = \mathcal{R}(N_1, m^0, P)\) implies that transitions \(\{t_i^j\} \in T_2\) are state-enabled at a marking if and only if \(P(m, t_i^j) = 1\) for some \(t_i \in T_{1c}\). To see this, assume \(\exists t_i^j \in T_2\) and a marking \(m\) such that \(t_i^j \in T_2\) and \(P(m, t_i^j) = 0\), where \(t_i \in T_{1c}\). Then \((m + C \times 1_i) \in \mathcal{R}(N_2, m^0) - \mathcal{R}(N_1, m^0, P)\), which is a contradiction. In the same way if \(\exists t_i \in T_{1c}\) and a marking \(m\) such that \(\nexists t_i^j \in T_2\) such that \(t_i^j \in T_2\) but \(P(m, t_i^j) = 1\), then \((m + C \times 1_i) \in \mathcal{R}(N_1, m^0, P) - \mathcal{R}(N_2, m^0)\), which is again a contradiction.

Therefore, \(\forall m, \forall t \in T_2, (t \in T_{1c} \land (m, t) = 1) \Leftrightarrow (P(m, t) = 1 \text{ for some } t \in T_1)\). A transition that is enabled at a marking is also enabled at all larger markings. This means that \(P\) is a marking-monotone policy. Since \(N_2\) is a \(B\)-preserving composition, it means that \(P\) is an MM-BESP.

\((\Leftarrow)\) We prove that the \(B\)-preserving composition of \(N_1\) and a \(B\)ESP \(P\). We use induction to prove that \(\mathcal{R}(N_2, m^0) = \mathcal{R}(N_1, m^0, P)\). The base case is the initial marking \(m^0\). Consider a string \(\sigma\) that is a valid firing string from \(m^0\). Suppose \(m^0 \xrightarrow{\sigma} m^1\) and \(m^0 \xrightarrow{\sigma} m^2\), where \(\sigma_i \in pr(\sigma)\). Here we use \(pr(\bullet)\) to denote the set of prefixes of the string argument. The induction hypothesis is that \(m^2 \in \mathcal{R}(N_2, m^0)\) and \(m^2 \in \mathcal{R}(N_1, m^0, P)\). Then from the construction in Algorithm 1, that \(\exists j\) such that \(t^j_i \in T_{1c}(N_2, m^1)\) (Step 8). Moreover, the firing of \(t^j_i\) adds \(c_i\)-many tokens in places in \(\Pi_2\), which is equal to the number of tokens added in \(\Pi_1\). Therefore, if \(m^1 \xrightarrow{\sigma_i} m^2\), then \(m^2 \in \mathcal{R}(N_2, m^0)\) and \(m^2 \in \mathcal{R}(N_1, m^0, P)\). On the other hand, if \(P(m^1, t_i) = 0\), then from Lemma 1, we have that \(m^i \geq \vec{m}\), for some \(\vec{m} \in \min(\mathcal{T}_M(B, N, t_i))\). Then it follows from the construction in Algorithm 1, that \(\exists j\) such that \(t^j_i \in T_{1c}(N_2, m^i)\) (Step 8). This constitutes the induction step.

An important consequence of the above theorem is that if there exists an MM-BESP \(P\) for \(N(m^0)\), then there exists a composition of \(\mathcal{P}\) and \(N(m^0)\). Moreover, due to the marking-monotone nature over the whole space of markings, the composition remains the same for any \(\vec{m} \geq m^0\). This is a desirable feature in the design of systems as analysis for various initial markings becomes easy, without having to evaluate the composition for each of them separately.

Suppose we want to synthesize a supervisory policy that enforces the property \(\Lambda^{i=1}_{c=1} B_c\) in a Petri net \(N_1(m^0)\). We assume that the existence of an MM-BESP, \(\mathcal{P}\), over reachable markings is decidable for all \(c \in \{1, \ldots, l\}\), and that there exists a procedure for synthesis. We also assume that the composed model and the supervisory policy that enforces \(\Lambda^{i=1}_{c=1} B_c\) is independent of the order in which \(B_c\)s are enforced. Algorithm 2 gives an outline of the procedure for the synthesis of a supervisory policy that enforces \(\Lambda^{i=1}_{c=1} B_c\). A
more specific procedure will depend on the properties that we want to enforce. Note that in Algorithm 2 we use the set $\gamma$ as a proxy for the MM policy, which due to result in Theorem 1 does not lead to any loss of generality.

Algorithm 2 SeqSYNTH($B_c, N_c$)

1: $\gamma = N^n$
2: while $(m^0 \in \gamma) \land \{ (B_i)_{i=1} \text{ are not enforced} \}$ do
3: for $c = 1$ to $l$ do
4: $\gamma = \text{SYNTHESIZE}((B_i, N_c))$
5: $N_{c+1} = \text{COMPOSE}(B_c, N_c)$
6: end for
7: end while

The algorithm stops either when the initial marking drops out of the estimate $\gamma$ or when property $\wedge_{c=1} B_c$ can be enforced by enforcing $\gamma$. However, whether a specific instance of the procedure (for given $\{ (B_i)_{i=1} \}$ will terminate or not will depend on the subroutine SYNTHESIZE. The subroutine SYNTHESIZE takes the current estimate of $\gamma$ and modifies it so that another property $B_c$ can be enforced. For several properties the whole loop in Algorithm 2 will terminate in a single iteration. But as discussed in Section I, a PN that is live can lose liveness if some markings that were originally reachable cannot be reached anymore. Therefore, if liveness is one of the $B_i$s, then the while loop in Algorithm 2 might have to execute several times till all the properties are enforced. Due to the typical nature of liveness, in the next section present a software tool for synthesizing an MM-LESP for an arbitrary PN $N$.

IV. AN ALGORITHM FOR MM-LESP SYNTHESIS

Suppose $m^0 \in \Omega$ and $P_\Omega$ is an MM-policy that enforces the control-invariant set $\Omega$. We can construct the coverability graph, $G(N(m^0), P_\Omega)$, of $N(m^0)$ under the supervision of $P_\Omega$, along the same lines as the coverability graph of a PN (cf. section 4.2.1, [6]). The policy $P_\Omega$ is an MM-ESD for $N(m^0)$ if and only if (i) $m^0 \in \Omega$, and (ii) there is a closed-path $v \Rightarrow v$ in $G(N(m^1), P_\Omega)$ for each $m^1 \in \text{min}(\Omega)$ such that all transitions appear at least once in $\sigma$, and the net-change in tokens in each place after the firing of $\sigma$ is non-negative (i.e. $\text{Cx}(\sigma) \geq 0$). References [9], [10] present the implementation details of an LESP synthesis algorithm for a class of PN structures using object-oriented concepts like encapsulation and polymorphism. Extending the LESP-synthesis code for MM-LESP synthesis required just a few modifications, which are described below.

The LESP-synthesis implementation was done within four major classes called PetriNet, NodeTable, MinimalElementsManager and MarkingVector, described in detail in [9] (cf. Figure 3). The PetriNet class has a member function doTheLoopTest() that checks for the path-requirement on the coverability graph as described in Figure 2. This test uses an ILP-based algorithm described in the appendix of [5] using the API of the MILP-solver lp_solve [11]. Among the other member functions of the PetriNet class is a member function computeMinimalElementsOfControlInvariantSet() which computes the (minimal elements of) largest control-invariant subset of any right-closed set of markings that was not control invariant with respect to $N$. This member function was modified, in the implementation for MM-LESP synthesis, to check if $m^0 \in \Omega_{i+1}$ before making the substitution of $\Omega_i \leftarrow \Omega_{i+1}$. The other classes (i.e. NodeTable, MinimalElementsManager and MarkingVector) and their member functions were kept with no change. The input and output structure of the earlier implementation was retained without change for the MM-LESP synthesis as well.
V. AN ILLUSTRATIVE EXAMPLE

We illustrate the procedure in Algorithm 1 and 2 using an example. We are interested in synthesizing a supervisory policy, $\mathcal{P}$, for the PN $N_1$ as shown in Figure 4. If $m^0$ is the initial marking, then the objectives for supervisions are: (1) $N_1(m^0)$ should be live; and (2) $\forall m^1 \in R(N_1, m^0, \mathcal{P})$, $m^1(p_6) + m^1(p_7) + m^1(p_8) \geq 1$. The set of initial markings for which an LESP exists, $\mathcal{D}(\mathcal{L}, N_1)$, is given by the right-closed set with minimal elements $\hat{m}_1 = (1 0 0 0 0 0 0 0)^T$ and $\hat{m}_2 = (0 0 0 1 1 0 0 0)^T$ (10). Let $C_1 = (-1 1 0 0 0 0 0 0)^T$ denote the column of the incidence matrix $C$ corresponding to transition $t_1$. Applying the result from Lemma 1, we get two minimal elements of $\mathcal{T}_M(\mathcal{L}, N_1, t_1)$, \{\( \hat{m}_i \}_{i=1,2}$, as: $\hat{m}_1 = \max(0, \hat{m}_1 - C_1) = (2 0 0 0 0 0 0 0)^T$ and $\hat{m}_2 = \max(0, \hat{m}_2 - C_1) = (1 0 0 1 0 0 0 0)^T$. Upon firing of $t_1$ from $\hat{m}_1$ and $\hat{m}_2$, we get the markings $\tilde{m}_1 = (1 1 0 0 0 0 0 0)^T$ and $\tilde{m}_2 = (0 1 0 1 1 0 0 0)^T$ respectively. The first thing to note is that $\tilde{m}_1, \tilde{m}_2 \in \mathcal{D}(\mathcal{L}, N_1)$. The PN $N_2$ which is a composition of $N_1$ and the LESP $\mathcal{P}_L$ is shown in Figure 5. Transition $t_1$ in $N_1$ is replaced by two new controllable transitions $t^1_1$ and $t^2_1$. The input (output) arc-weights of $t^1_1$ and $t^2_1$ correspond to $\tilde{m}_1$ and $\tilde{m}_2$ (resp. $\tilde{m}_1$ and $\tilde{m}_2$) respectively. Transition $t_2$ will always be enabled by the LESP $\mathcal{P}_L$. Therefore, there is no change in it. It can be verified that $R(N_2, m^0) = R(N_1, m^0, \mathcal{P}_L)$. Next we synthesize an LESP that enforces the property: $\forall m^1 \in R(N_1, m^0, \mathcal{P}), m^1(p_6) + m^1(p_7) + m^1(p_8) \geq 1$. Let $\mathcal{D}_M(\mathcal{C}, N_1) = \{m \in \mathcal{N}^n : m(p_6) + m(p_7) + m(p_8) \geq 1\}$. Since the tokens in place $p_8$ can be lost by the uncontrollable firing of $t_{10}$, transition $t_8$ should be controlled enabled if and only if the resulting marking is in $\mathcal{D}_M(\mathcal{C}, N_1)$. By Lemma 1, the minimal elements of $\mathcal{T}_M(\mathcal{C}, N_1, t_1)$ are: \{\( (0 0 0 0 0 0 2 0)^T, (0 0 0 0 1 1 0)^T \)\}. For this particular example, the intersection of $\mathcal{D}_M(\mathcal{L}, N_1)$ and $\mathcal{D}_M(\mathcal{C}, N_1)$ gives us an estimate of the set of markings for which a supervisory policy that enforces both properties exists.

VI. CONCLUSION

In this paper, we formalized the concept of composition for a PN model and supervisory policy. A PN model of the supervised system was obtained by systematically replacing a controllable transition in the plant model by a set of transitions. We proved that there exists a composition of a PN model and a supervisory policy if and only if the supervisory policy is marking monotone over its reachable markings. We also presented a procedure for obtaining a Petri net model of the supervised system. One of the open problems is to identify different interpretations of composition and to obtain similar results.

REFERENCES