Decidability and Closure Properties of Weak Petri Net Languages in Supervisory Control

Alessandro Giua,

Dip. di Ingegneria Elettrica ed Elettronica, Università di Cagliari,

Piazza d'Armi, 09123 Cagliari, Italy

Phone: +39-070-675-5892 – Fax: +39-070-675-5900 – Email: giua@diee.unica.it

Frank DiCesare

Dept. of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute

Troy NY 12180-3590, USA – Email: dicesare@ecse.rpi.edu

Abstract

We extend the class of control problems that can be modeled by Petri nets considering the notion of weak terminal behavior. Deterministic weak languages represent closed-loop terminal behaviors that may be enforced by nonblocking Petri net supervisors if controllable. The class of deterministic weak PN languages is not closed under the supremal controllable sublanguage operator.

Published as:
A. Giua, F. DiCesare, "Decidability and Closure Properties of Weak Petri Net Languages in Supervisory Control," *IEEE Trans. on Automatic Control*, Vol. 40, No. 5, pp. 906–910, May, 1995.

1 Introduction

In this note we present a notion of terminal behaviors for Petri nets (PN) called *weak behaviors* and study their use in Supervisory Control Theory (SCT) [10]. Weak behaviors overcomes some of the problems due to the use of Petri net *marked behaviors* [2, 15, 13].

Since the weak behavior is specifically defined only for Petri nets, we will introduce it with an example.

In Figure 1, we have shown a simple communication process where a sender S sends messages to a receiver R through an infinite-capacity channel C. The initial state of the system is shown in the figure. A token in S_on (S_off) means that the sender is active (has disconnected). A token in R_on (R_off) means that the receiver is active (has disconnected). The tokens in C represent messages sent on the channel but not yet received.

We consider as final marking M_f, with $M_f(\text{S\_off}) = M_f(\text{R\_off}) = 1$ and $M_f(\text{S\_on}) = M_f(\text{C}) = M_f(\text{R\_on}) = 0$. We may consider as *terminal behavior* of the net its *marked language*, i.e., the set of all firing sequences that reach the final marking M_f. Since $M_f(\text{C}) = 0$, this means that in a terminal state there may not be tokens in C, i.e., all messages sent by S have been received by R. However, we may consider as terminal behavior of the net its *weak language*, i.e., the set of all firing sequences that reach a marking *greater or equal to* M_f. This means that we consider as terminal all those states in which both sender and receiver have disconnected, i.e., $M_f(\text{S\_off}) = M_f(\text{R\_off}) = 1$, regardless of the number of tokens contained in C.

The choice between weak and marked language as terminal behavior depends on the physical problem. For the example we considered here, accepting the weak language as terminal behavior means that we are not interested in ensuring that all messages sent are received, or that we are not capable of enforcing this constraint. It may be interesting to note that a Petri net is, in effect, a weak counter, in the sense that if a transition firing may occur at a marking M then it may also occur at any marking $M' > M$ (see Lemma 2.1). Thus it also make sense to assume that if a marking M is final then any marking $M' > M$ is also final.

This note discusses the use of weak languages in supervisory control. In particular, since deterministic weak PN languages have not been studied before, we also devoted some time to the study of their properties. Our main results can be summarized as follows. Firstly, the classes of weak and marked languages generated by deterministic nets are incomparable. Thus, taking also into account the weak behavior of deterministic nets (in addition to the marked behavior) we extend the class of control problems that can be modeled by PN. Secondly, deterministic weak PN languages are DP-closed, i.e., they represent closed-loop terminal behaviors that may be enforced by Petri net supervisors. This is an important result, that does not hold for the class of deterministic marked PN languages. Thirdly, the main properties of interest in supervisory control, such as controllability and L-closure, are decidable when this class of languages is considered. It is also decidable whether a system is weakly blocking. Finally, the class of deterministic weak PN languages is not closed under the supremal controllable sublanguage operator.

2 Background

2.1 Petri Nets

A *Place/Transition net* (P/T net) is a structure $N = (P, T, Pre, Post)$ where: P is a set of *places*; T is a set of *transitions*; $Pre : P \times T \to \mathbb{N}$ specifies the arcs directed from places to transitions; $Post : P \times T \to \mathbb{N}$ specifies the arcs directed from transitions to places. Here $\mathbb{N} = \{0, 1, 2, \ldots\}$. See [7, 11] for a more complete definition of Petri nets.

A *marking* is a vector $M : P \to \mathbb{N}$. $\mathbb{N}^{|P|}$ will denote the set of all possible markings that may be defined on the net. A *P/T system* or *net system* $\langle N, M_0 \rangle$ is a net N with an initial marking M_0. We will expand sometimes the definition of marking to a function: $M : P \to \mathbb{N}_\omega$, with $\mathbb{N}_\omega = \mathbb{N} \cup \{\omega\}$. ω is a new element such that for all $n \in \mathbb{N} : n < \omega$, and for all $n \in \mathbb{N}_\omega : \omega + n = n + \omega = \omega - n = \omega$.

When a marking M' can be reached from marking M by executing a *firing sequence* of transitions $\sigma = t_1 \ldots t_k$ we write $M\ [\sigma\rangle\ M'$. We write $M\ [\sigma\rangle$ to denote that σ may be executed from M. The set of markings reachable on a net N from a marking M is called the *reachability set* of M and is denoted as $R(N, M)$.

A *labeled Petri net* (or *Petri net generator*) [4] is a 4-tuple $G = (N, \ell, M_0, F)$ where: $N = (P, T, Pre, Post)$ is a Petri net structure; $\ell : T \to \Sigma$ is a labeling function that assigns to each transition a label from the alphabet of events Σ and will be extended to a mapping $T^* \to \Sigma^*$ in the usual way; M_0 is an initial marking; F is a finite set of final markings. The finiteness of F is essential in the definition of labeled nets. We also define the *covering set* of F as $C_F = \{M \in \mathbb{N}_\omega^{|P|} \mid (\exists M' \in F)\ [M \geq M']\}$.

We will represent a discrete event system (DES) as a labeled Petri net. Given a DES $G = (N, \ell, M_0, F)$, the *L-type language* of G (called *marked behavior* in the framework of SCT) is $L_m(G) = \{\ell(\sigma) \in \Sigma^* \mid \sigma \in T^*, M_0\ [\sigma\rangle\ M, M \in F\}$; the *G-type language* of G (that we will call *weak behavior*) is $L_w(G) = \{\ell(\sigma) \in \Sigma^* \mid \sigma \in T^*, M_0\ [\sigma\rangle\ M, M \in C_F\}$;

the *P-type language* of G (called *closed behavior* in the framework of SCT) is $L(G) = \{\ell(\sigma) \in \Sigma^* \mid \sigma \in T^*, M_0 [\sigma\rangle\}$.

Note that in our definition of labeled net we are assuming that ℓ is a λ-free labeling function, i.e., no transition is labeled with the empty string λ and two (or more) transitions may have the same label. The classes of L-type, G-type, and P-type languages generated by λ-free labeled nets are denoted $\mathcal{L}$, $\mathcal{G}$, and $\mathcal{P}$ respectively.

A *deterministic* PN generator [4] is such that the string of events generated from the initial marking uniquely determines the marking reached. Formally, a DES G is deterministic iff for all $t, t' \in T$, with $t \neq t'$, and for all $M \in R(N, M_0)$: $M [t\rangle \wedge M [t'\rangle \Longrightarrow [\ell(t) \neq \ell(t')] \vee [Post(\cdot, t) - Pre(\cdot, t) = Post(\cdot, t') - Pre(\cdot, t')]$. Note that we are slightly extending the definition of determinism used by [4, 13]. In fact, we accept as deterministic a system in which two transitions with the same label may be simultaneously enabled at a marking M, provided that the two markings reached from M by firing t and t' are the same (it is a kind of parallelism of transitions). Our definition will preserve the properties of deterministic nets, such as the fundamental Lemma 2.2 in the following.

Systems of interest in SCT are deterministic, hence we will always assume that the generators considered here are deterministic. The classes of L-type, G-type, and P-type PN languages generated by deterministic PN generators are denoted $\mathcal{L}_d$, $\mathcal{G}_d$, and $\mathcal{P}_d$. We also define [2] the class of *deterministic prefix closed* (DP-closed for short) PN languages as $\mathcal{L}_{DP} = \{L \in \mathcal{L} \mid \overline{L} \in \mathcal{P}_d\}$.

We also recall the closure properties for the class $\mathcal{G}_d$. Proofs of these properties can be found in [3].

Definition 2.1. *Given a language $L \subseteq \Sigma^*$, its complement language is $\mathbf{C}L = \Sigma^* \setminus L$. Given a class of languages $\mathcal{A}$, we denote co-$\mathcal{A} = \{\mathbf{C}L \mid L \in \mathcal{A}\}$ the class of all the complements of languages in $\mathcal{A}$.*

Proposition 2.1. *(a) The class co-$\mathcal{G}_d$ is included in $\mathcal{L}$. (b) The class co-$\mathcal{G}_d$ is not included in $\mathcal{G}_d$. (c) The class $\mathcal{G}_d$ is closed under intersection. (d) The class $\mathcal{G}_d$ is not closed under union.*

2.2 Languages and Control

Let L be a language on alphabet Σ. Its *prefix closure* is the set of all prefixes of strings in L: $\overline{L} = \{\sigma \in \Sigma^* \mid \exists \tau \in \Sigma^* \ni \sigma\tau \in L\}$. A language L is said to be *closed* if $L = \overline{L}$. If $K, L \subset \Sigma^*$ are languages, K is said to be *L-closed* if $K \cap L = \overline{K} \cap L$.

Let in the following G be a DES with alphabet of events Σ. G is *nonblocking* if any string that belongs to its closed behavior may be completed to a string that belongs to its marked behavior. A deterministic G is nonblocking iff $L(G) = \overline{L_m(G)}$.

The alphabet of events Σ is partitioned into two disjoint subsets: Σ_c, the set of *controllable events*, and Σ_u, the set of *uncontrollable events*. A language $K \subset \Sigma^*$ is said to be *controllable* with respect to $L(G)$ [10] if $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$. The set of all languages controllable with respect to $L(G)$ is denoted $\mathcal{C}(G)$.

If a language $L \subset \Sigma^*$ is not controllable with respect to $L(G)$ we may compute its *supremal controllable sublanguage* [10] defined as: $L^\uparrow = \sup\{K \subseteq L \mid K \in \mathcal{C}(G)\}$.

2.3 Known results on PN

We present here some lemmas that will be used in the following.

Lemma 2.1 ([11], Lemma 5.2c). *Let $\langle N, M_0 \rangle$ be a marked Petri net, M_1, M_2 be markings of N with $M_1 \leq M_2$, and σ be a sequence of transitions. Then $M_1[\sigma\rangle M_1'$ implies $M_2[\sigma\rangle M_2'$ and $M_1' \leq M_2'$.*

Lemma 2.2 ([9], P1). *Let $G = (N, \ell, M_0, F)$ be a deterministic PN generator and $w \in L(G)$. There exists one and only one marking M such that $(\exists\sigma)$ $[M_0[\sigma\rangle M, \ell(\sigma) = w]$.*

Lemma 2.3 (*Lemma of Dickson*, [16]). *Let $A \subset I\!N^k, k \geq 1$, be an infinite set of vectors of length k and $B \subseteq A$ the set of minimal vectors for the ordering $\leq$ defined by: $a \leq a'$ (with $a, a' \in A$) iff $a(i) \leq a'(i)$ for all $i = 1, \ldots, k$. Then B is finite.*

3 Deterministic Languages

In this section we study the relationships among different classes of deterministic PN languages.

For the classes of λ-free PN languages the following strict inclusions hold: $\mathcal{P} \subset \mathcal{G} \subset \mathcal{L}$ [4]. Hence if we consider as terminal language of a net its marked behavior, we can also model all other kinds of λ-free PN languages as terminal languages.

In the case of deterministic nets, however, it is possible to prove that $\mathcal{G}_d$ and $\mathcal{L}_d$ are incommensurable.

Proposition 3.1. *(a) The class $\mathcal{P}_d$ is strictly included in $\mathcal{G}_d$. (b) The class $\mathcal{L}_d$ is not included in $\mathcal{G}_d$. (c) The class $\mathcal{P}_d$ is not included in $\mathcal{L}_d$. (d) The class of regular languages $\mathcal{R}$ is included in $\mathcal{G}_d \cap \mathcal{L}_d$.*

Proof:

(a) It is a well known result, since the P-type language of a generator G is also a G-type language for the same net with set of final markings $F = \{\vec{0}\}$. The inclusion is strict, since all languages in $\mathcal{P}_d$ are prefix closed, while languages in $\mathcal{G}_d$ need not be.

(b) In [8] it was shown that the language $L = \{a^m b^m \mid m > 0\}$ (that is clearly in $\mathcal{L}_d$) is not a weak PN language.

(c) Consider the language $L = \{a^m b^n \mid m \geq n \geq 0\}$ (that is clearly in $\mathcal{P}_d$). We will prove that L is not in $\mathcal{L}_d$.

Assume that L is the marked behavior of the generator $G = (N, \ell, M_0, F)$. Let M_i be the unique marking (by Lemma 2.2) such that $M_0[\sigma_i\rangle M_i$ with $\ell(\sigma_i) = a^i$. We prove that $M_j \neq M_{j'}$ if $j \neq j'$. In fact, assume $M_j = M_{j'}$ with $j > j'$. Since $a^j b^j \in L$, then there exist σ'_j such that $\ell(\sigma'_j) = b^j$, $M_j[\sigma'_j\rangle M'$ and $M' \in F$. However, since $M_{j'} = M_j$, $M_{j'}[\sigma'_j\rangle M'$ and $M' \in F$, i.e., $a^{j'} b^j \in L(G)$, clearly a contradiction. This shows that there are infinitely many markings M_i. But all these markings must be final markings, since $a^i \in L$ for all i. This contradicts the hypothesis that F is a finite set.

(d) It is a well known result, since for any finite state automaton there exists an equivalent (i.e., generating the same languages) state machine Petri net whose initial marking consists of a single token in the place corresponding to the initial state. Note, furthermore, that the marked and weak languages of such a net are the same. We conjecture that the reverse inclusion holds, i.e., we conjecture that $\mathcal{R} = \mathcal{G}_d \cap \mathcal{L}_d$ [1]. $\diamond$

A Venn diagram of deterministic PN languages is shown in Figure 2. The shaded area contains (possibly is identical to) the class of regular languages.

The previous proposition shows that both classes $\mathcal{L}_d$ and $\mathcal{G}_d$ are a strict superset of regular languages, and that they represent incomparable classes of languages. Thus, taking into account the weak behavior (in addition to the marked behavior) of deterministic nets we extend the class of control problems that can be modeled by PN.

If our conjecture that $\mathcal{R} = \mathcal{G}_d \cap \mathcal{L}_d$ is correct, we may see the marked and weak languages of Petri nets as two inherently different ways of extending the power of regular languages. In the case of marked languages, we extend to encompass the languages generated by deterministic PN generators with an infinite state space but an *always finite* set of final markings. In the case of weak languages, we extend to encompass the languages generated by deterministic PN generators with an infinite state space and an *always infinite* set of final markings.

From the point of view of supervisory control, no great changes are required to take into account weak behaviors. We just need to extend the notion of nonblockingness and of controlled behavior to weak languages.

A DES G is *weakly nonblocking* if any string that belongs to its closed behavior may be completed to a string that belongs to its weak behavior. A deterministic G is weakly nonblocking iff $L(G) = \overline{L_w(G)}$.

Assume we use Petri nets to represent both a nonmarking supervisor S and a system G to control. The *closed behavior* and the *controlled behavior* of the closed-loop system have been defined as [10]: $L(S/G) = L(G) \cap L(S)$, and $L_m(S/G) = L_m(G) \cap L(S/G) = L_m(G) \cap L(S)$. We may now define the *controlled weak behavior* of the closed-loop system as: $L_w(S/G) = L_w(G) \cap L(S/G) = L_w(G) \cap L(S)$. The supervisor S is weakly nonblocking if $L(S/G) = \overline{L_w(S/G)}$.

4 Supervisors for Weak Languages

The class $\mathcal{L}_{DP}$ of DP-closed PN languages was defined in [2]. DP-closed languages represent closed-loop terminal behaviors that may be enforced by nonblocking Petri net supervisors as the next theorem [2] implies.

Theorem 4.1. *Let G be a nonblocking PN and let $L \subseteq L_m(G)$ be a nonempty language. There exists a nonblocking PN supervisor S such that $L_m(S/G) = L$ iff $L \in \mathcal{L}_{DP}$, L is controllable, and L is $L_m(G)-$closed.*

When the weak behavior of a net is used as terminal behavior we have a similar result.

Theorem 4.2. *Let G be a weakly nonblocking PN and let $L \subseteq L_w(G)$ be a nonempty language. There exists a weakly nonblocking PN supervisor S such that $L_w(S/G) = L$ iff $L \in \mathcal{L}_{DP}$, L is controllable, and L is $L_w(G)-$closed.*

Proof: The proof is substantially identical that of the previous theorem as given in [2]. $\diamond$

In [2] was also shown that $\mathcal{L}_d \not\subseteq \mathcal{L}_{DP}$, i.e., not all deterministic marked languages represent closed-loop terminal behaviors that may be enforced by nonblocking Petri net supervisors. The next theorem proves that all deterministic weak languages are DP-closed.

Theorem 4.3. *The class $\mathcal{G}_d$ is included in $\mathcal{L}_{DP}$.*

Proof: We will prove a slightly stronger property, namely that given a weakly blocking deterministic PN generator $G = (N, \ell, M_0, F)$ there exists a finite procedure to construct a new deterministic PN generator G' such that $L_w(G') = L_w(G)$ and $L(G') = \overline{L_w(G')}$.

For all transitions t of G, we define two disjoint sets:

$$\mathcal{M}_e^t = \{M \in I\!N^{|P|} \mid M[t\rangle M', R(N, M') \cap C_F \neq \emptyset\};$$

$$\mathcal{M}_d^t = \{M \in I\!N^{|P|} \mid M[t\rangle M', R(N, M') \cap C_F = \emptyset\}.$$

$\mathcal{M}_e^t$ is the set of all markings of N from which the firing of t does not lead to a blocking marking. $\mathcal{M}_d^t$ is the set of all markings of N from which the firing of t leads to a blocking marking. Generally, both sets may be infinite.

Let $\underline{\mathcal{M}_e^t} \subseteq \mathcal{M}_e^t$ be the set of minimal markings of $\mathcal{M}_e^t$ for the ordering $\leq$. This set is finite by the Lemma of Dickson (Lemma 2.3), and we will show later that it can be computed with a finite procedure. Using the monotonicity property of PN, as in Lemma 2.1, it is easy to prove that for all $M_e \in \underline{\mathcal{M}_e^t}$ and for all $M_d \in \mathcal{M}_d^t$, $M_d \not\geq M_e$, i.e., no marking in $\mathcal{M}_d^t$ may be greater or equal to a marking in $\underline{\mathcal{M}_e^t}$. Let $\underline{\mathcal{M}_e^t} = \{M_1, \ldots, M_k\}$. We may remove transition t from the net, adding a set of new transitions $T_t = \{t_1, \ldots, t_k\}$, such that for all $t_i \in T_t$: $\ell(t_i) = \ell(t)$, $Pre(\cdot, t_i) = M_i$, $Post(\cdot, t_i) = M_i - Pre(\cdot, t) + Post(\cdot, t)$. No transition in T_t may fire from a marking $M_d \in \mathcal{M}_d^t$, since for all $t_i \in T_t$, $M_d \not\geq M_i = Pre(\cdot, t_i)$. Hence, all firings of t leading to a blocking marking are now prevented. However, there exists a $t_i \in T_t$ firable from any marking $M_e \in \mathcal{M}_e^t$, since by definition of minimal marking set there exist a marking $M_i \in \underline{\mathcal{M}_e^t}$ such that $M_e \geq M_i = Pre(\cdot, t_i)$. Once the construction is repeated for transitions $t \in T$, we have a new PN generator G' such that $L_w(G') = L_w(G)$ and $L(G') = \overline{L_w(G')}$.

Finally, we will prove that G' is deterministic. The only nondeterminism may arise from the fact that we have introduced (at each step) a set of transitions T_t with the same label, and there may exist a reachable marking $M_e \in \mathcal{M}_e^t$ such that there exists two enabled transitions $t_i, t_j \in T_t$, with $i \neq j$. However, by construction $Post(\cdot, t_i) - Pre(\cdot, t_i) = Post(\cdot, t_j) - Pre(\cdot, t_j)$ hence determinism is preserved. $\diamond$

The construction presented in the previous theorem is similar to the one presented in [16] (Construction 4.1). We will use a result of [16] to prove that the set $\underline{\mathcal{M}_e^t}$ can effectively be computed. First we give two definitions.

Definition 4.1. *A set $\mathcal{M} \subseteq I\!N^k$ is a* right-closed set *if $\{M \in I\!N^k \mid (\exists M' \in \mathcal{M})\ [M \geq M']\} \subseteq \mathcal{M}$.*

Definition 4.2. *A set $\mathcal{M} \subseteq I\!N^k$ has the* property RES *if for all $M_\omega \in I\!N_\omega^k$ it is decidable whether $\{M \in I\!N^k \mid M \leq M_\omega\} \cap \mathcal{M} \neq \emptyset$.*

Using these definitions we can state the next lemma.

Lemma 4.1. *Let $\mathcal{M} \subseteq I\!N^k$ be a* right-closed set. *Then the set $\underline{\mathcal{M}}$ of minimal markings of $\mathcal{M}$ for the ordering $\leq$ can be effectively constructed iff $\mathcal{M}$ has property RES.*

The proof of the lemma is given in [16] by showing an algorithm to compute the set $\underline{\mathcal{M}}$.

Proposition 4.1. *Given a net system $\langle N, M_0 \rangle$, with $N = (P, T, Pre, Post)$, for all $t \in T$ the set $\mathcal{M}_e^t = \{M \in I\!N^{|P|} \mid M[t\rangle M', R(N, M') \cap C_F \neq \emptyset\}$ is right-closed and has property RES.*

Proof: Using Lemma 2.1 it is immediate to prove that $\mathcal{M}_e^t$ is right-closed. We prove that this set has also property RES. Fix a $M_\omega \in I\!N_\omega^{|P|}$. We may decide if $M_\omega \in \mathcal{M}_{e,\omega}^t = \{M \in I\!N_\omega^{|P|} \mid M[t\rangle M', R(N, M') \cap C_F \neq \emptyset\}$. In fact: 1) Clearly it is decidable if $M_\omega \geq Pre(\cdot, t)$. 2) Let M'_ω be the marking such that $M_\omega[t\rangle M'_\omega$; by the analysis of the coverability graph [11] of $\langle N, M'_\omega \rangle$ we may decide if $R(N, M'_\omega) \cap C_F \neq \emptyset$ (we need to verify that there exists a node in the coverability graph labeled by a marking covering some marking in F).

Finally, there exists a finite marking $M \leq M_\omega$, with $M \in \mathcal{M}_e^t$, if and only if $M_\omega \in \mathcal{M}_{e,\omega}^t$, since $Pre(\cdot, t)$ and the markings in F are finite vectors. $\diamond$

From the previous lemma and proposition, it follows that the set $\underline{\mathcal{M}_e^t}$, defined in Theorem 4.3, can be effectively computed as shown in [16].

5 Decidability

In this section we discuss the decidability of some properties of discrete event systems studying the corresponding languages.

It is well known [12] that the inclusion problem: "Is $L_1 \subseteq L_2$?", with $L_1, L_2 \in \mathcal{P}$, is undecidable. However, the emptiness problem: "Is $L = \emptyset$?", with $L \in \mathcal{L}$, is decidable since it may be reduced to the reachability problem, shown to be decidable [5, 6, 12].

For deterministic PN languages the following lemma holds.

Lemma 5.1. *For $L_1 \in \mathcal{L}$ and $L_2 \in \mathcal{L}_d \cup \mathcal{G}_d$ it is possible to decide if $L_1 \subseteq L_2$.*

Proof: Pelz [9] noted that if $\mathbf{C}L_2 \in \mathcal{L}$ the inclusion problem may be reduced to the emptiness problem for the language $L = L_1 \cap \mathbf{C}L_2 \in \mathcal{L}$. She also proved that co-$\mathcal{L}_d \subseteq \mathcal{L}$. By Proposition 2.1.a we know that co-$\mathcal{G}_d \subseteq \mathcal{L}$. $\diamond$

Using this lemma, we now show that three important properties, weakly blockingness, L-closure, and controllability are decidable for deterministic systems.

Proposition 5.1. *It is possible to decide if a deterministic Petri net generator G is weakly blocking.*

Proof: We need to prove that it is possible to decide if $L(G) = \overline{L_w(G)}$. Since $L(G) \supseteq \overline{L_w(G)}$, we just need to show that $L(G) \subseteq \overline{L_w(G)}$ is decidable. By Theorem 4.3, $\overline{L_w(G)} \in \mathcal{P}_d \subseteq \mathcal{G}_d$ and the result follows from Lemma 5.1. $\diamond$

We cannot use the same reasoning to prove that it is possible to decide if G is blocking. In fact, in [2] we showed that $\overline{L_m(G)}$ is not necessarily a PN language.

Proposition 5.2. *It is possible to decide if a language $K \in \mathcal{L}_{DP}$ is controllable with respect to a Petri net generator G.*

Proof: We need to prove that it is possible to decide if $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$. Since $K \in \mathcal{L}_{DP}$, then $\overline{K} \in \mathcal{P}_d \subseteq \mathcal{G}_d$. Also $\overline{K}\Sigma_u \cap L(G) \in \mathcal{L}$ ($\mathcal{L}$ is closed under concatenation and intersection). Hence the result follows from Lemma 5.1. $\diamond$

Proposition 5.2 extends a result presented by Sreenivas [13]. Sreenivas showed that the controllability of a closed *free-labeled* language K with respect to a free-labeled PN generator G is decidable. His proof, however, applies to deterministic closed languages as well.

Note also that Sreenivas has discussed two different notions of controllability [14]. The strongest notion of controllability requires that we may also test for the inclusion: $K \subseteq L$. In this case it is necessary that G be a deterministic generator, as in the next proposition.

Proposition 5.3. *It is possible to decide if a language $K \in \mathcal{L}_{DP}$ is controllable with respect to a deterministic Petri net generator G and is contained in $L(G)$.*

Proof: Proposition 5.2 implies that controllability is decidable. The containment $K \subseteq L(G)$ is also decidable (by Lemma 5.1) since $L(G) \in \mathcal{P}_d \subseteq \mathcal{G}_d$. $\diamond$

Our final result regards the decidability of L-closure.

Proposition 5.4. *For all $L \in \mathcal{L}$, it is possible to decide if a language $K \in \mathcal{G}_d \cup (\mathcal{L}_d \cap \mathcal{L}_{DP})$ is L-closed.*

Proof: We need to prove that it is possible to decide if $\overline{K} \cap L = K \cap L$. Since $\overline{K} \cap L \supseteq K \cap L$, we just need to show that $\overline{K} \cap L \subseteq K$ is decidable. Since $K \in \mathcal{G}_d \cup (\mathcal{L}_d \cap \mathcal{L}_{DP})$ then $\overline{K} \cap L \in \mathcal{L}$, and the result follows from Lemma 5.1. $\diamond$

A remark on the complexity of these decision procedures. Assume we have two nets G_1 and G_2 whose closed (or marked or weak) behaviors are the languages L_1 and L_2. As suggested by the proof of Lemma 5.1, to check whether $L_1 \subseteq L_2$ we may follow these steps:

1. construct a PN G_2' generating $\mathbf{C}L_2$ (this is possible if G_2 is deterministic);
2. construct the net G as the intersection of the nets G_1 and G_2';
3. check whether the language generated by G is empty.

The first step may be carried out with the construction shown in [9], whose complexity has not been computed. The second step may be done efficiently. We expect the last step to have the same complexity of checking the reachability of a given marking, that is at best decidable in exponential space [4]. Reutenauer [12] has noted that the proof of decidability of the reachability problem due to Mayr [6] and Kosaraju [5] does not provide efficient algorithms for verifying whether a marking is reachable or not.

6 Supremal Controllable Sublanguage for Weak Languages

$\mathcal{G}_d$ is not closed under the supremal controllable sublanguage operator as next example shows.

Example 6.1. Let G be the PN generator in Figure 3, with $\Sigma_u = \{a\}$, $M_0 = (1000)^T$, and set of final markings $F = \{(0001)^T\}$. Now consider the net E in Figure 4 with set of final markings $F = \{(0001)^T\}$. The language $L_w(E) \in \mathcal{G}_d$ is not controllable. In Figure 5 we have shown the reachability tree of the two nets; since E refines G, we have represented the arcs that belong to the reachability tree of both nets with continuous lines, while the arcs that only belong to the reachability tree of G have been represented by dotted lines. $L_w(E)$ is not controllable because of the presence of the dotted arcs associated to the uncontrollable transition a. Applying the $^\uparrow$ operator, we obtain the supremal controllable sublanguages $L_w(E)^\uparrow = \{a^m b a^m (bc)^* b \mid m \geq 0\}$, that is the marked behavior of the generator in Figure 6. $L_w(E)^\uparrow \notin \mathcal{L}_{DP}$, as shown in [2], hence $L_w(E)^\uparrow \notin \mathcal{G}_d$.

Note 6.1. Whenever the language $L^\uparrow$ is not a weak PN language, we may be tempted to consider the supremal element of the class: $\mathcal{C}_g(L) = \{K \mid K \subseteq L, K \text{ is controllable}, K \in \mathcal{G}_d\}$. Note, however, that this supremal element does not always exist. In fact, the existence and uniqueness of the supremal controllable sublanguage in [10] follows from the fact that the class of controllable languages is closed under arbitrary union, while the class of weak languages is not closed under union (see Proposition 2.1.d). In the previous example, for instance, for all $i \in I\!N$ the language $K_i = \{a^n b a^n (bc)^* b \mid n \leq i\}$ is in $\mathcal{C}_g(L_w(E))$ and $K_i \subset K_{i+1}$. As we have seen in the example, $K_\infty = L_w(E)^\uparrow$ is not in $\mathcal{G}_d$.

7 Conclusions

We have studied a class of terminal behaviors of place/transition nets, called *weak behaviors.*

The classes of weak and marked languages generated by deterministic nets are incomparable. Thus, taking also into account the weak behavior of deterministic nets (in addition to the marked behavior) we extend the class of control problems that can be modeled by PN.

Deterministic weak PN languages are DP-closed, i.e., they represent closed-loop terminal behaviors that may be enforced by nonblocking Petri net supervisors. This is an important result that does not hold for the class of deterministic marked PN languages. The main properties of interest in supervisory control, such as controllability and L-closure, are decidable when this class of languages is considered. It is also decidable whether a system is weakly blocking. The complexity of these decision procedures has to be further studied. The class of deterministic weak PN languages is not closed under the supremal controllable sublanguage operator.

References

[1] A. Giua, F. DiCesare, "Weak Petri Net Languages for Supervisory Control," *Proc. 32nd IEEE Conf. on Decision and Control* (San Antonio, Texas), pp. 229–234, December, 1993.

[2] A. Giua, F. DiCesare, "Blocking and Controllability of Petri Nets in Supervisory Control," *IEEE Trans. on Automatic Control*, Vol. 39, No. 4, pp. 818–823, April, 1994.

[3] A. Giua, "On the Closure Properties of Deterministic Weak Petri Net Languages," *Technical Report No. 59*, Istituto di Elettrotecnica, University of Cagliari (Italy), February, 1994.

[4] M. Jantzen, "Language Theory of Petri Nets," and "Complexity of Place/Transition Nets," *Advances in Petri Nets 1986*, LNCS No. 254-I, pp. 397–412, Springer-Verlag, 1987.

[5] S.R. Kosaraju, "Decidability of Reachability in Vector Addition Systems," *Proc. 14th Ann. ACM Symp. on Theory of Computing* (San Francisco, California), pp. 267–281, May, 1982.

[6] E.W. Mayr, "An Algorithm for the General Petri Net Reachability Problem," *SIAM J. of Computing*, Vol. 13, No. 3, pp. 441–460, August, 1984.

[7] T. Murata, "Petri Nets: Properties, Analysis and Applications," *Proceedings IEEE*, Vol. 77, No. 4, pp. 541–580, April, 1989.

[8] M. Parigot, E. Pelz, "A Logical Formalism for the Study of Finite Behaviour of Petri Nets," *Advances in Petri Nets 1985*, LNCS No. 222, pp. 346–361, Springer-Verlag, 1986.

[9] E. Pelz, "Closure Properties of Deterministic Petri Net Languages," *Proc. STACS 1987*, LNCS No. 247, pp. 373–382, Springer-Verlag, 1987.

[10] P.J. Ramadge, W.M. Wonham, "The Control of Discrete Event Systems," *Proceedings IEEE*, Vol. 77, No. 1, pp. 81–98, January, 1989.

[11] W. Reisig, "Petri Nets: An Introduction," Springer-Verlag, 1985.

[12] C. Reutenauer, "The Mathematics of Petri Nets," Masson and Prentice-Hall Intern., 1990.

[13] R.S. Sreenivas, "A Note on Deciding the Controllability of a Language K with respect to a Language L," *IEEE Trans. on Automatic Control*, Vol. 38, No. 4, pp. 658–662, April, 1993.

[14] R.S. Sreenivas, "On a Weaker Notion of Controllability of a Language K with respect to a Language L," *IEEE Trans. on Automatic Control*, Vol. 38, No. 9, pp. 1446–1447, September, 1993.

[15] R.S. Sreenivas, B.H. Krogh, "On Petri Net Models of Infinite State Supervisors," *IEEE Trans. on Automatic Control*, Vol. 37, No. 2, pp. 274–277, February, 1992.

[16] R. Valk, M. Jantzen, "The Residue of Vector Sets with Applications to Decidability Problems in Petri Nets," *Acta Informatica*, No. 21, pp. 643–674, Springer-Verlag, 1985.

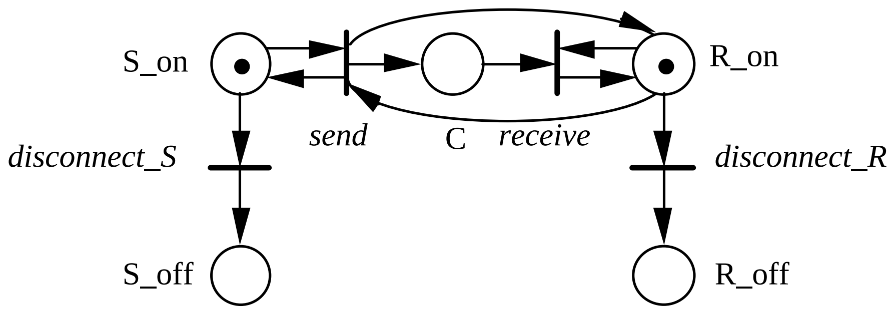

Figure 1: A sender-receiver process.

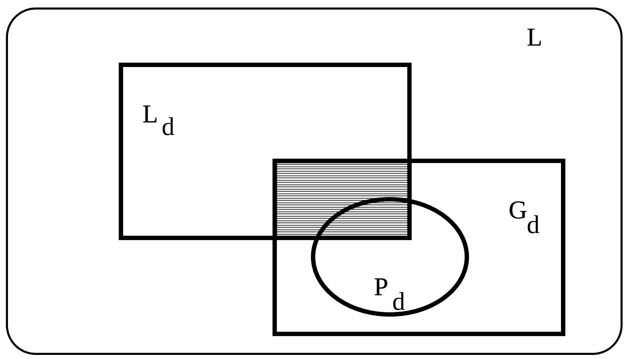

Figure 2: Venn diagram of deterministic Petri net languages.

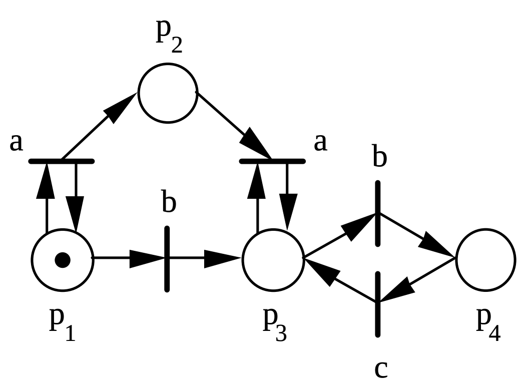

Figure 3: Generator G in Example 6.1.

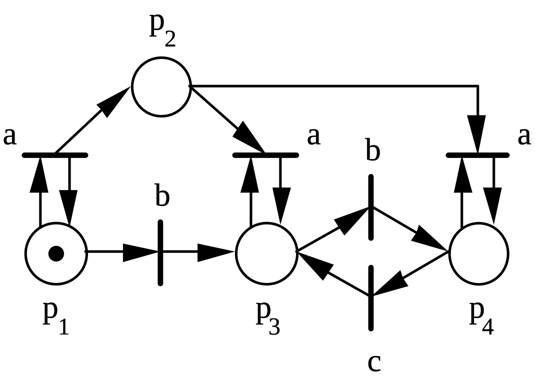

Figure 4: Generator E in Example 6.1.

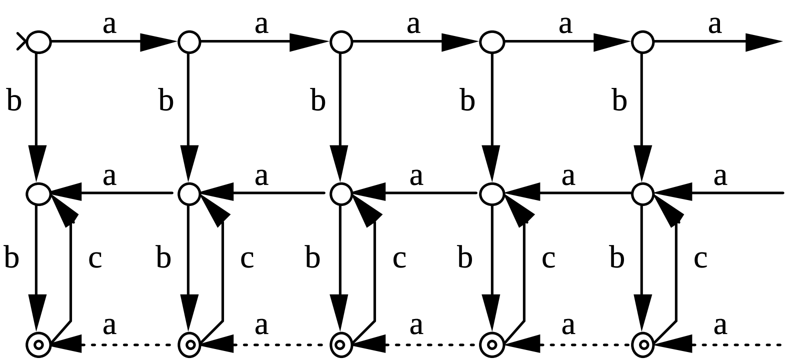

Figure 5: Reachability tree of G and E in Example 6.1.

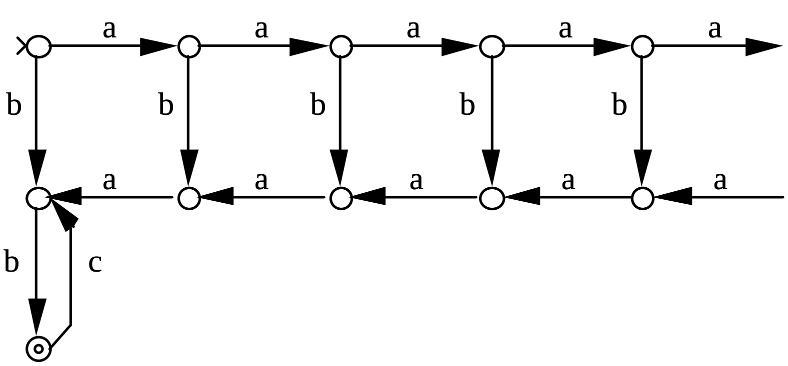

Figure 6: Generator of $L_w(E)^{\uparrow}$ in Example 6.1.