A New Approach for Diagnosability Analysis of Petri Nets Using Verifier Nets

Maria Paola Cabasino, Alessandro Giua, Stéphane Lafortune, Carla Seatzu*‡

Abstract

In this paper we analyze the diagnosability properties of labeled Petri nets. We consider the standard notion of diagnosability of languages, requiring that every occurrence of an unobservable fault event be eventually detected, as well as the stronger notion of diagnosability in K steps, where the detection must occur within a fixed bound of K event occurrences after the fault. We give necessary and sufficient conditions for these two notions of diagnosability for both bounded and unbounded Petri nets and then present an algorithmic technique for testing the conditions based on linear programming. Our approach is novel and based on the analysis of the reachability/coverability graph of a special Petri net, called *Verifier Net*, that is built from the Petri net model of the given system. In the case of systems that are diagnosable in K steps, we give a procedure to compute the bound K. To the best of our knowledge, this is the first time that necessary and sufficient conditions for diagnosability and diagnosability in K steps of labeled unbounded Petri nets are presented.

Published as:
M.P. Cabasino, A. Giua, S. Lafortune, C. Seatzu, "A New Approach for Diagnosability Analysis of Petri Nets using Verifiers Nets," *IEEE Trans. on Automatic Control*, Vol. 57, No. 12, pp. 3104 - 3117, Dec 2012. The original publication is available at www.ieeexplore.ieee.org.

*This work has been partially supported by the European Community's Seventh Framework Programme under project DISC (Grant Agreement n. INFSO-ICT-224498) and by the US National Science Foundation (Grants ECCS-0624281 and CNS-0930081).

†M.P. Cabasino and A. Giua and C. Seatzu are with the Department of Electrical and Electronic Engineering, University of Cagliari, Piazza D'Armi, 09123 Cagliari, Italy {cabasino,giua,seatzu}@diee.unica.it

‡S. Lafortune is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Mi 48109-2122, USA stephane@eecs.umich.edu

1 Introduction

In this paper we present new results on the analysis and testing of the diagnosability properties of labeled Petri nets. A labeled Petri net dynamic system is diagnosable if every occurrence of an unobservable fault transition can be detected within a finite number of transition firings, based on observed transition labels. We consider the situation where two or more transitions of the net may share the same observed label. In addition to the unobservable fault transitions, there may be other transitions of the net that are unobservable. Our novel analysis technique is applicable to both bounded and unbounded Petri nets; a Petri net is unbounded when the token count of one or more places can become arbitrarily large. Furthermore, in addition to the notion of diagnosability where the detection delay of each fault occurrence is finite for every sequence of transition firings, we also consider the stronger notion of diagnosability in K steps, where the detection delay is uniformly bounded by K over all sequences of transition firings. We give necessary and sufficient conditions for both notions of diagnosability. These conditions are applicable to both bounded and unbounded nets. We also present a procedure to test for these necessary and sufficient conditions. Finally, we briefly discuss two methods from the literature to perform online diagnosis of bounded and unbounded Petri nets.

To the best of our knowledge, this is the first time that necessary and sufficient conditions for diagnosability and diagnosability in K steps of labeled Petri nets with unbounded state spaces are presented. Note that, as shown in the paper, the two above notions of diagnosability coincide in the case of bounded systems. The results in this paper therefore provide a way to check diagnosability and diagnosability in K steps for discrete event systems that generate languages that are not necessarily regular. Most of the results on diagnosability analysis of discrete event systems are focused on systems modeled by finite-state automata, i.e., systems that generate regular languages. Unbounded Petri nets can generate languages that are not regular.

The paper is organized as follows. In Section 2, we give an overview of relevant literature on diagnosability analysis and online diagnosis of Petri nets, and contrast our contributions with respect to these works. In Section 3, we present necessary background on labeled Petri nets. In Section 4, we recall the standard algorithm for the construction of the coverability graph of a Petri net and illustrate some properties of this graph. In Section 5, we formally state and compare the two notions of diagnosability considered in the paper, and then we prove that they coincide in the case of Petri nets whose sequences of transition firings generate regular languages. In Section 6, we present the development of the necessary and sufficient conditions for diagnosability and diagnosability in K steps. First we present an algorithm for the construction of a special Petri net, called *verifier net*, built form the original Petri net under consideration; it is called a "verifier" net because it bears some similarity with the verifier automata used in the study of the diagnosability of discrete event systems modeled by finite-state automata. Then, we present necessary and sufficient conditions for diagnosability and diagnosability in K steps based on the analysis of the reachability/coverability graph of the verifier net. Finally, we give a procedure to compute the bound K in the case of systems that are diagnosable in K steps. In Section 7, we describe a test for diagnosability of bounded and unbounded Petri nets in terms of the necessary and sufficient conditions of Section 6. In Section 8, we suggest two approaches that

can be used for online diagnosis of bounded and unbounded Petri net systems, where solving an online diagnosis problem means associating to each observed string of events a diagnosis state, such as "normal", "faulty" or "uncertain". Section 9 concludes the paper. A preliminary and partial version of this paper was presented in [5].

2 Literature review

Fault diagnosis in dynamic systems is a subject that has received a lot of attention in the past decades. In the context of discrete event systems (DES), several original approaches have been proposed using *automata* models; see [4, 12, 21, 27, 26, 22, 28, 34, 35, 36, 18] for a sample of this work. Automata models often suffer from the problem of combinatorial explosion of the state space, when the system is composed of several interacting components. Petri nets provide compact representations of systems that exhibit concurrency, and their structural analysis may provide a way to overcome the combinatorial explosion problem. However, analytical approaches based on the reachability/coverability graph of the Petri net, as in this paper, do not in general mitigate the combinatorial explosion problem. Several diagnosis approaches have been proposed for Petri net models of DES; see [31, 37, 17, 3, 42, 25, 32, 13, 2, 15, 33] for relevant references. These past works are primarily focused on the problem of online diagnosis, and none of them provides conditions or a procedure for determining *a priori* if a given system is *diagnosable*, in the sense of the language-based definition of diagnosability introduced in [35].

In the last few years, some results have been presented for diagnosability of *bounded* Petri nets ([29, 8, 16, 23]). In particular, the work in [29] presents an approach to verify diagnosability in the framework of Petri net unfoldings based on the twin plant method. It consists in constructing a verifier, which compares pairs of paths from the initial model sharing the same observable behavior. The construction of the verifier net in [29] is based on similar ideas to those discussed in our paper, although used in the context of safe Petri nets and with several differences due to a different technical approach.

In [8], some of us presented an original approach for diagnosability analysis of bounded Petri nets based on the notion of basis markings that allows to reduce the state space enumeration. Unfortunately, in the case of unbounded Petri nets, the basis marking approach cannot be used because in such a case we need an exhaustive enumeration of the nodes of the coverability graph. On the other hand, when applicable, the approach in [8] may be preferable to the approach in the present paper, because it allows to solve the online diagnosis problem using the same framework as for diagnosability analysis.

In the case of infinite state systems, some results have been presented lately in the framework of (unbounded) Petri nets. The first contribution on diagnosability of unbounded Petri nets was given by Ushio *et al.* in [38]. That work extends the necessary and sufficient condition for diagnosability given by Sampath *et al.* [35, 36] to unbounded Petri nets. It is assumed that the set of places is partitioned in observable and unobservable places, while all transitions are unobservable in the sense that their occurrences cannot be observed. Starting from the Petri

net, they build a diagnoser called *simple ω diagnoser* that provides sufficient conditions for diagnosability of unbounded Petri nets. In [11], in contrast with [38], it is assumed that some of the transitions of the Petri net are observable and shown that the additional information from observed transitions in general enhances the diagnosability of the analysed system. Moreover, starting from the diagnoser, Chung proposes an automaton called *verifier* that allows a polynomial check mechanism on diagnosability, but for bounded Petri nets only. Another relevant work is [40] where Wen and Jeng propose an approach to test diagnosability by checking the structural property of T-invariant of the net. They use the diagnoser of [38] to prove that their method is correct, however they do not construct a diagnoser for the system to do online diagnosis. In [41], Wen *et al.* present a linear-programming-based algorithm of polynomial complexity in the number of nodes for computing a sufficient condition of diagnosability of DES modeled by Petri nets.

Our problem statement is related to prior works on diagnosability analysis of regular languages represented by finite-state automata and is related to but different from the above-described prior works on diagnosability analysis of Petri nets. Specifically, we adopt a language-based approach, where only the labels of the transition firings are observed; token counts in places are not observable, except for the initial system state. Moreover, we consider labeled Petri nets where two or more transitions can share the same label, rather than so-called "free-labeled" Petri nets. We also note that our approach is applicable to both bounded and unbounded nets. In the case of bounded Petri net systems, we will show that our methodology for diagnosability analysis based on the verifier net provides an alternative to the usual approach of building an automaton from the (finite) reachability graph of the Petri net and then applying the diagnoser methodology of [35] or the verifier methodology of [43] for instance.

3 Background on Petri nets

In this section we present the necessary background and define the notation used in the paper. For more details on Petri nets, we refer the reader to [30].

A *Place/Transition net* (P/T net) is a structure $N = (P, T, Pre, Post)$ where: P is a set of m places; T is a set of n transitions; and $Pre : P \times T \to \mathbb{N}$ and $Post : P \times T \to \mathbb{N}$ are the *pre*– and *post*– incidence functions that specify the arcs. The incidence matrix C of the net is equal to $C = Post - Pre$. A *marking* (i.e., net state) is a vector $M : P \to \mathbb{N}$ that assigns to each place of a P/T net a nonnegative integer number of tokens, represented by black dots in diagrams. We denote by $M(p)$ the marking of place p. A P/T *system* or *net system* $\langle N, M_0 \rangle$ is a net N with an initial marking M_0. Hereafter we refer to a P/T net as a Petri net, often abbreviated as PN.

A transition t is said to be enabled at M iff $M \geq Pre(\cdot, t)$; an enabled transition t may fire yielding the marking $M' = M + C(\cdot, t)$. We write $M\,[\sigma\rangle$ to denote that the sequence of transitions $\sigma = t_{j_1} \cdots t_{j_k}$ is enabled at M, and we write $M\,[\sigma\rangle\,M'$ to denote that the firing of σ yields M'. The set of all finite sequences that are enabled at the initial marking M_0 is denoted by $L(N, M_0)$, i.e., $L(N, M_0) = \{\sigma \in T^* \mid M_0[\sigma\rangle\}$. We use λ to denote an empty sequence of transitions, i.e.,

$\sigma\lambda = \lambda\sigma = \sigma, \quad \forall \quad \sigma \in T^*$. Given a sequence $\sigma \in T^*$, we call $\pi : T^* \to \mathbb{N}^n$ the function that associates to σ a vector $y \in \mathbb{N}^n$, named the *firing vector (or Parikh vector)* of σ. Specifically, $y = \pi(\sigma)$ is such that $y(t) = k$ if the transition t is contained k times in σ.

A marking M is *reachable* in $\langle N, M_0 \rangle$ if there exists a firing sequence σ such that $M_0 \ [\sigma\rangle \ M$. The set of all markings reachable from M_0 defines the *reachability set* of $\langle N, M_0 \rangle$ and is denoted by $R(N, M_0)$. Finally, we denote by $PR(N, M_0)$ the *potentially reachable set*, i.e., the set of all markings $M \in \mathbb{N}^m$ for which there exists a vector $y \in \mathbb{N}^n$ that satisfies the *state equation* $M = M_0 + C \cdot y$, i.e., $PR(N, M_0) = \{M \in \mathbb{N}^m \mid \exists\ y \in \mathbb{N}^n \ : \ M = M_0 + C \cdot y\}$. We have that $R(N, M_0) \subseteq PR(N, M_0)$.

A Petri net is said *ordinary* if $Pre, Post \in \{0, 1\}^{m \times n}$, i.e., if each arc has weight equal to one. A *state machine* is an ordinary PN where each transition has exactly one input and one output place. A Petri net having no directed circuits is called *acyclic*. For this subclass, it can be shown that the state equation gives necessary and sufficient conditions for reachability [30].

A net system $\langle N, M_0 \rangle$ is *bounded* if there exists a positive constant k such that, for all $M \in R(N, M_0)$, $M(p) \leq k$. The finite reachability space is represented by a graph called *reachability graph* (RG). If the number of tokens in one or more places can grow arbitrarily large, then the Petri net system is *unbounded* and the graph representing the infinite state space is called *coverability graph* (CG).

Definition 3.1 *Given a Petri net system $\langle N, M_0 \rangle$, a transition t is:*

- dead *if there does not exist a reachable marking $M \in R(N, M_0)$ that enables t;*
- semi-live *if there exists at least one marking $M \in R(N, M_0)$ that enables t;*
- live *if for each reachable marking $M \in R(N, M_0)$, t is semi-live in $\langle N, M \rangle$.* ■

A net system $\langle N, M_0 \rangle$ is *live* if all transitions $t \in T$ are live. A *deadlock* occurs at marking M if no transition is enabled at M.

Definition 3.2 *A sequence $\sigma \in T^*$ is called* repetitive *if there exists a marking $M_1 \in R(N, M_0)$ such that*

$$M_1[\sigma\rangle M_2[\sigma\rangle M_3[\sigma\rangle \cdots \tag{1}$$

i.e., if it can fire infinitely often starting from M_1. It is possible to distinguish two different types of repetitive sequences:

- stationary *sequence: if in* (1) *$M_i = M_{i+1}$ for all $i = 1, 2, \ldots$.*
- increasing *sequence: if in* (1) *$M_i \lneq M_{i+1}$ for all $i = 1, 2, \ldots$.* ■

An example is provided in the next section.

There exists a simple structural condition to characterize repetitive sequences:

Fact 3.3 *[30] If sequence σ is enabled at M_1, a necessary and sufficient condition for σ to be repetitive is that in (1), $M_i \leq M_{i+1}$ for all $i = 1, 2, \ldots$, or, equivalently $C \cdot y \geq \vec{0}$, where $y = \pi(\sigma)$. Furthermore if $C \cdot y = \vec{0}$ the sequence is stationary, else if $C \cdot y \gneq \vec{0}$ it is increasing.* ■

Observe that any sequence of infinite length contains a repetitive sequence. This is a trivial consequence of a result proved in [24] for an infinite sequence of vectors. A nonnegative integer vector $y \in \mathbb{N}^n$ satisfying $C \cdot y = 0$ is called a *T-invariant*.

A *labeling function* $\mathcal{L} : T \to L \cup \{\varepsilon\}$ assigns to each transition $t \in T$ either a symbol from a given alphabet L or the empty string ε. We call *labeled Petri net system* the triple $\langle N, M_0, \mathcal{L} \rangle$. We denote by T_u the set of transitions whose label is ε, i.e., $T_u = \{t \in T \mid \mathcal{L}(t) = \varepsilon\}$. Transitions in T_u are called *unobservable* or *silent*. We denote by T_o the set of transitions labeled with a symbol in L. Transitions in T_o are called *observable* because when they fire their label can be observed. In this paper we assume that the same label $l \in L$ can be associated with more than one transition. In particular, two transitions $t_1, t_2 \in T_o$ are called *indistinguishable* if they share the same label, i.e., $\mathcal{L}(t_1) = \mathcal{L}(t_2) = l \in L$.

We extend the labeling function to define the *projection operator* $\mathcal{L} : T^* \to L^*$ recursively as follows:

(i) if $t_j \in T_o$ then $\mathcal{L}(t_j) = l$ for some $l \in L$;

(ii) if $t_j \in T_u$ then $\mathcal{L}(t_j) = \varepsilon$;

(iii) if $\sigma \in T^* \wedge t_j \in T$ then $\mathcal{L}(\sigma t_j) = \mathcal{L}(\sigma)\mathcal{L}(t_j)$.
Moreover, $\mathcal{L}(\lambda) = \varepsilon$, where λ is the empty sequence of transitions.

Using the extended labeling function, the language of transition labels is therefore denoted by $\mathcal{L}(L(N, M_0))$. We use the notation w for a string of transition labels, i.e., $w = \mathcal{L}(\sigma)$ where σ is a transition sequence. Note that the length of a sequence σ (denoted by $|\sigma|$) is always greater than or equal to the length of the corresponding string w (denoted by $|w|$). In fact, if σ contains k' transitions labeled with ε, then $|\sigma| = k' + |w|$. The *inverse projection operator* $\mathcal{L}^{-1}$ is defined as $\mathcal{L}^{-1}(w) = \{\sigma \in L(N, M_0) : \mathcal{L}(\sigma) = w\}$. Given a language $K \subseteq T^*$, we denote by K/s the post-language of K after s, i.e., $K/s = \{g \in T^* \mid sg \in K\}$. We say that K is *prefix-closed* if all the prefixes of all the strings in K are also in K.

We conclude this section with the following definition:

Definition 3.4 *Given a net $N = (P, T, Pre, Post)$, and a subset $T' \subseteq T$ of its transitions, we define the $T'-$*induced subnet *of N as the new net $N' = (P, T', Pre', Post')$ where $Pre', Post'$ are the restrictions of $Pre, Post$ to T'. In this case, we write $N' \prec_{T'} N$.* ■

The net N' can be thought of as being obtained from N by removing all transitions in $T \setminus T'$ and all dangling arcs.

4 Coverability graph

One technique frequently used for the analysis of unbounded Petri nets is based on the construction of the coverability tree/graph (see, e.g., [30]) that provides a description in finite terms of the infinite reachability set. For the sake of completeness, we review briefly the construction of this graph. Each node of the coverability graph is labeled with an m–dimensional row vector whose entries may either be an integer number or may be equal to the special symbol ω, while arcs are elements of T and are labeled by $(t, \mathcal{L}(t))$ if a transition labeling function has been defined. The symbol ω denotes that the marking of the corresponding place may grow infinitely large. Note that, for all $n \in \mathbb{N}$, we have that $\omega > n$ and $\omega \pm n = \omega$.

Algorithm 4.1 ***Construction of the coverability tree for*** $\langle N, M_0 \rangle$***.***

1. *Label the root node q_0 with the initial marking M_0 and mark it "new".*
2. ***While** a node tagged "new" exists **do***
   (a) *Select a node q marked "new" and let M be its tag.*
   (b) ***For** all t enabled at M, i.e., such that $M \geq Pre(\cdot, t)$:*
      i. *Let $M' = M + C(\cdot, t)$ be the marking reached from M by firing t.*
      ii. *Let $\bar{q}$ be the first node met on the backward path from q to q_0 whose label is $\bar{M} \leq M'$. **If** such a node exists **then** for all $p \in P$ such that $M'(p) > \bar{M}(p)$ let $M'(p) = \omega$.*
      iii. *Add a new node q' and tag it M'.*
      iv. *Add an arc labeled t (or $(t, \mathcal{L}(t))$) from q to q'.*
      v. ***If** there already exists a node with tag M' in the tree, **then** tag node q' "duplicate", **else** tag it "new".*
   (c) *Untag node q.* ■

From the coverability tree (CT) one can obtain the coverability graph (CG) by fusing duplicate nodes with the untagged node with the same label; one can always convert a CT into a CG, and vice-versa.

In the construction of the CT the existence of a sequence σ that leads from a marking $\bar{M}$ to a greater marking M' is identified at step 2.(*b*).*ii*. The places that grow unbounded by the repeated firing of such a sequence σ are assigned a special symbol ω. Note that if $\bar{M}$ contains no ω-places, then σ is an increasing sequence. However, if $\bar{M}$ contains ω-places we can only say that σ is increasing for all places p such that $\bar{M}(p) < \omega$ and $M'(p) = \omega$: nothing can be said for the remaining places. Marking containing ω-places will be denoted in the following as ω-markings. We write $\mathbb{N}_\omega$ the result of $\mathbb{N} \cup \{\omega\}$.

Given a marking $M \in \mathbb{N}^m$, let us say that M is ω-covered by $M_\omega \in \mathbb{N}_\omega^m$ if $M_\omega(p) = M(p)$ for each place p such that $M_\omega(p) \neq \omega$. Let us denote this by $M_\omega \geq_\omega M$. The CG gives us only

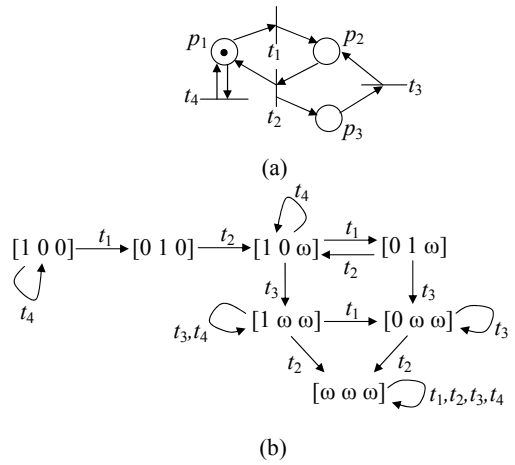

Figure 1: (a) Petri net in Example 4.3 and (b) its coverability graph.

necessary conditions for reachability, namely a marking M is reachable only if there exists at least one marking M_ω that *ω-covers* M. Now we state a property of the CG that will be useful in the following development.

Proposition 4.2 *[30] Let us consider a PN system and its CG. If a transition t is in the CG then t is* semi-live. ■

This means that if a transition appears in the CG, then there exists for sure a firing sequence that enables it.

Let us consider a path in a graph. If the initial and the terminal vertices of this path coincide, then the path is called a *cycle*. A cycle is called *elementary* if no vertex appears more than once in it.

We conclude this section with an example to illustrate the notions introduced in this and the preceding sections.

Example 4.3 *Let us consider the Petri net in Fig. 1.(a) and its CG in Fig. 1.(b). This Petri net is unbounded. It has two different repetitive sequences. The sequence $\sigma_1 = t_1 t_2$ is a repetitive increasing sequence; indeed, its firing increases the number of tokens in place p_3. The sequence $\sigma_2 = t_4$ is a repetitive stationary sequence; its firing does not change the number of tokens in the places of the net. Note that there are cycles in the CG, containing markings with ω in some places, that are not associated with a repetitive sequence. As an example, transition t_3 will never be able to fire infinitely often from any reachable marking, even though it corresponds to a cycle in the CG.* ■

5 Diagnosability of Petri net systems

We are interested in diagnosability analysis of potentially unbounded Petri nets. In this regard, we make the following assumptions.
(A1) The structure of the net and its initial marking M_0 are known.
(A2) Two or more observable transitions may share the same label.
(A3) The system does not enter a deadlock after the firing of any fault transition.
The last assumption is a weakened version of the usual "liveness" assumption in most works on diagnosability of DES; it avoids the technicalities that must be dealt with when the system may deadlock after a fault. We comment further on this assumption at the end of this section.

The property of diagnosability is commonly defined in terms of observed strings of events. Bounded PNs necessarily generate regular languages, as the reachability graph provides a finite-state automaton representation of the language. This result applies to the language of transition sequences (i.e, over T^*), as well as to the language of transition labels (i.e., over L^*), since the latter is obtained from the former by a labeling map which is a homomorphism (see, e.g., [19]). On the other hand, the language of transition sequences (as well as the language of their labels) generated by an unbounded PN may or may not be a regular language (an example is presented later in this section). To properly handle the case of languages that are not regular, we need a definition of diagnosability that is slightly different from the one introduced in [35] concerning the diagnosability of regular languages. Before presenting the definition we have adopted, we need to introduce some further notation.

As was mentioned in Section 3, when an observable transition $t \in T_o$ occurs, we observe its label $\mathcal{L}(t) \in L$. Unobservable transitions (those in T_u) yield the empty symbol ε. For the purpose of diagnosability, the set of unobservable transitions is partitioned into two subsets, namely $T_u = T_f \cup T_{reg}$ where T_f includes all fault transitions (modeling anomalous or faulty behavior), while T_{reg} includes all transitions pertaining to unobservable but "regular" events. The set T_f is further partitioned into r different subsets T_f^i, where $i = 1, \ldots, r$, model the different fault classes. Let T' be a subset of T. We define $\Psi(T') = \{st' \in L(N, M_0) : t' \in T'\}$, i.e., the set of all firing sequences in $L(N, M_0)$ that end with a transition $t' \in T'$.

We consider the following definition of diagnosability of Petri nets inspired by the definition of diagnosability for (regular) languages introduced in [35].

Definition 5.1 *A labeled Petri net system $\langle N, M_0, \mathcal{L}\rangle$ having no deadlock after the occurrence of any transition $t_f \in T_f^i$, for $i \in \{1, \ldots, r\}$, is* diagnosable with respect to (w.r.t.) the fault class T_f^i *if*

$$\begin{aligned} &\forall s \in \Psi(T_f^i), \quad \exists K_s \in \mathbb{N}, \quad \forall g \in L(N, M_0)/s, \\ &|g| \geq K_s \;\Rightarrow\; \forall w \in \mathcal{L}^{-1}(\mathcal{L}(sg)),\; \exists t_f \in T_f^i \;:\; t_f \in w. \end{aligned} \tag{2}$$

Note that the bound K_s may depend on the particular string s.

A labeled Petri net system $\langle N, M_0, \mathcal{L}\rangle$ is said to be diagnosable *if it is diagnosable w.r.t. all fault classes.* ■

In words, a labeled Petri net system $\langle N, M_0, \mathcal{L}\rangle$ having no deadlock after the occurrence of any transition $t_f \in T_f^i$, for $i \in \{1, \ldots, r\}$, is *not diagnosable with respect to the fault class* T_f^i if there exist two firing sequences σ_1 and $\sigma_2 \in T^*$ satisfying the following four conditions for any $k \in \mathbb{N}$:

- $\mathcal{L}(\sigma_1) = \mathcal{L}(\sigma_2)$, i.e., the sequences have the same observable projection;
- $\sigma_1 \in (T \setminus T_f^i)^*$, i.e., σ_1 does not contain a fault transition in the fault class T_f^i;
- there exists at least one fault transition $t_f \in T_f^i$ such that $t_f \in \sigma_2$,
- σ_2 is of "arbitrary length" after fault $t_f \in T_f^i$, i.e., there exists at least one decomposition $\sigma_2 = \sigma_2' t_f \sigma_2''$ with $|\sigma_2''| > k$.

Let us now define the notion of diagnosability in K steps.

Definition 5.2 *[35] A labeled Petri net system $\langle N, M_0, \mathcal{L}\rangle$ having no deadlock after the occurrence of any transition $t_f \in T_f^i$, for $i \in \{1, \ldots, r\}$ is* diagnosable in (at most) K steps w.r.t. the fault class T_f^i *if* $\exists K \in \mathbb{N}$ *such that*

$$\begin{aligned} & \forall s \in \Psi(T_f^i), \quad \forall g \in L(N, M_0)/s, \\ |g| \geq K \Rightarrow \quad & \forall w \in \mathcal{L}^{-1}(\mathcal{L}(sg)),\ \exists t_f \in T_f^i \ : \ t_f \in w. \end{aligned} \tag{3}$$

A labeled Petri net system $\langle N, M_0, \mathcal{L}\rangle$ is said to be diagnosable in K steps *if it is diagnosable in K steps w.r.t. all fault classes.* ■

The above definition means that a Petri net system having no deadlock after the occurrence of any transition $t_f \in T_f^i$, for $i \in \{1, \ldots, r\}$, is diagnosable in K steps w.r.t. the $i-$th fault class if for any sequence s that terminates in a transition in T_f^i and for any continuation g of s of length greater than or equal to K, all sequences w having the same observable projection as sg contain some fault transition in T_f^i. In other words, diagnosability in K steps w.r.t. a given fault class implies that the occurrence of a fault in that class can be detected after the finite number K of transition firings.

The key point here is that in diagnosability in K steps (Definition 5.2), there exists a bound K for the detection delay after the fault event that is *uniform* over all sequences of transition firings. In contrast, in the definition of diagnosability (Definition 5.1), there need not exist a uniform bound. This distinction, unnecessary in the case of languages generated by finite-state automata, is now needed in the case of potentially non-regular languages, where the detection delay could grow arbitrarily large. Consider the following result.

Proposition 5.3 *(i) A labeled PN system $\langle N, M_0, \mathcal{L}\rangle$ that is* diagnosable in K steps w.r.t. T_f^i *is also* diagnosable w.r.t. T_f^i.

(ii) If the language $L(N, M_0)$ is regular, then the converse to (i) also holds.

Proof *Definition 5.2 is obviously stronger than Definition 5.1 because it requires that the bound K should be the same for all strings $s \in \Psi(T_f^i)$; hence the first part of the statement holds.*

Assume now that $L(N, M_0)$ is a regular language, hence can be generated by a finite-state automaton with transition function $\delta : X \times T \to X$ over finite state space X, and with initial state x_0. It is not difficult to see that if $\langle N, M_0, \mathcal{L}\rangle$ is diagnosable w.r.t. T_f^i, for any two strings $s, s' \in \Psi(T_f^i)$ with $\delta(x_0, s) = \delta(x_0, s')$, one may choose the same integer K_s in (2), i.e., K_s actually depends on the state $\delta(x_0, s)$. Since there is a finite number of states, by taking the largest K over all states reached by strings in $\Psi(T_f^i)$, we conclude that $\langle N, M_0, \mathcal{L}\rangle$ is diagnosable in K steps w.r.t. T_f^i. □

In the above proposition we used regularity of the language $L(N, M_0)$, not that of the language of transition labels $\mathcal{L}(L(N, M_0))$. We make two important remarks.

Remark 5.4 *The regularity of $L(N, M_0)$ is decidable; see [39]. The same result does not hold, in general, for $\mathcal{L}(L(N, M_0))$; see [20].* ■

Remark 5.5 *Part (ii) of Proposition 5.3 is no longer valid if stated in terms of the regularity of language $\mathcal{L}(L(N, M_0))$; we present a counter-example for this situation. Consider the Petri net in Fig. 2, where $T_o = \{t_1, t_4, t_5, t_7, t_9\}$, $T_u = \{\varepsilon_2, \varepsilon_3, \varepsilon_6, \varepsilon_8\}$ and $T_f = \{\varepsilon_2\}$. Let $\mathcal{L}(t_1) = \mathcal{L}(t_4) = \mathcal{L}(t_5) = a$, $\mathcal{L}(t_7) = d$ and $\mathcal{L}(t_9) = c$. The prefix-closed language*

$$\begin{aligned} L(N, M_0) = PC[\{\varepsilon\} \cup \{t_1^\alpha \varepsilon_2 t_4^\beta \varepsilon_6 t_7^\gamma \mid \alpha, \beta, \gamma \geq 0 \wedge \beta \leq \alpha\} \\ \cup \{t_1^\alpha \varepsilon_3 t_5^\beta \varepsilon_8 t_9^\gamma \mid \alpha, \beta, \gamma \geq 0 \wedge \beta \leq \alpha\}] \end{aligned}$$

where PC stands for the operation of taking the prefix closure, is not *regular. This can be easily shown using the pumping lemma*[1] *in [19]. Using the notation in the footnote, let $w = t_1^n \varepsilon_2 t_4^n \in L(N, M_0)$, where $x = t_1^{n-\delta}$, $y = t_1^\delta$, with $1 \leq \delta \leq n$, and $z = \varepsilon_2 t_4^n$. We have that $|xy| = |t_1^n| = n$, $|y| = |t_1^\delta| \geq 1$, but $w' = xy^i z$ for $i = 0$ does not belong to the language, i.e., $t_1^{n-\delta} \varepsilon_2 t_4^n \notin L(N, M_0)$. This means that the language $L(N, M_0)$ is not regular. On the contrary, the labeled language* is *regular:*

$$\mathcal{L}(L(N, M_0)) = \{a^*(c^* + d^*)\}.$$

Here, the net is diagnosable, but it is not diagnosable in K steps. For all strings $s(k) \in \Psi(T_f)$, where $s(k) = t_1^k \varepsilon_2$, in (2) one may choose $K_{s(k)} = k + 1$ or greater to prove that the system is diagnosable. Since this value of $K_{s(k)}$, however, grows arbitrarily large with k, the system is not diagnosable in K steps for any finite K. Note that if we modify the label of transition t_7 as $\mathcal{L}(t_7) = c$ then this system is not diagnosable with respect to either definitions of diagnosability. ■

The second part of Proposition 5.3 shows that in the case of regular languages, it is not necessary to distinguish between the two notions of diagnosability. This result was also observed in [44] in the context of automata models.

The next example, which will be used as a running example in the remainder of this paper, shows a simplification of the unbounded net in Fig. 2 that is also diagnosable but not diagnosable in

[1] Let L be a regular language. Then there exists a constant n such that if w is any word in L, and $|w| \geq n$, we may write $w = xyz$ such that $|xy| \leq n$, $|y| \geq 1$ and for all $i \geq 0$ $xy^i z$ is in L. Furthermore, n is no greater than the number of states of the smallest finite automaton accepting L.

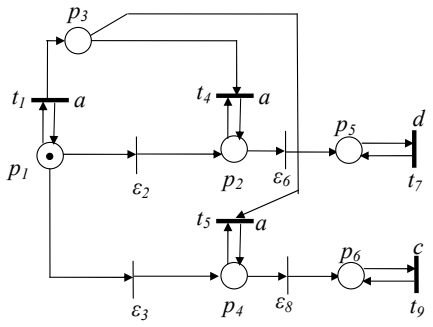

Figure 2: The Petri net system of Remark 5.5.

K steps. Recall that for the net in Fig. 2, the language $L(N, M_0)$ is not regular, while the labeled language $\mathcal{L}(L(N, M_0))$ is regular. On the other hand, in the example that follows, both languages (unlabeled and labeled) are non-regular.

Example 5.6 *Let us consider the Petri net system in Fig. 3, where* $T_o = \{t_1, t_4, t_5, t_6, t_7\}$, $T_u = \{\varepsilon_2, \varepsilon_3\}$ *and* $T_f = \{\varepsilon_2\}$. *Let* $\mathcal{L}(t_1) = a$, $\mathcal{L}(t_4) = \mathcal{L}(t_6) = b$, $\mathcal{L}(t_5) = d$ *and* $\mathcal{L}(t_7) = c$.

Using the same argument as in Remark 5.5, it is straightforward to verify that this net is diagnosable but not diagnosable in K *steps. Moreover, we note that neither*

$$L(N, M_0) = PC[\{\varepsilon\} \cup \{t_1^\alpha \varepsilon_2 \sigma \ \mid \ \alpha \geq 0,\ \sigma \in \{t_4, t_5\}^*,$$
$$|\sigma|_{t_4} \leq \alpha\} \ \cup \ \{t_1^\alpha \varepsilon_3 \sigma \ \mid \ \alpha \geq 0,\ \sigma \in \{t_6, t_7\}^*,\ |\sigma|_{t_6} \leq \alpha\}]$$

nor the labeled language

$$\mathcal{L}(L(N, M_0)) = PC[\{a^\alpha w \ \mid \ \alpha \geq 0,\ w \in \{b, d\}^*,$$
$$|w|_b \leq \alpha\} \ \cup \ \{a^\alpha w \ \mid \ \alpha \geq 0,\ w \in \{b, c\}^*,\ |w|_b \leq \alpha\}]$$

is regular. This can be easily shown using again the pumping lemma. In the case of $L(N, M_0)$, *we can choose* $w = t_1^n \varepsilon_2 t_4^n$, *where* $x = t_1^{n-\delta}$, $y = t_1^\delta$, *with* $1 \leq \delta \leq n$, *and* $z = \varepsilon_2 t_4^n$ *and using the same arguments as in Remark 5.5 we can show that it does not satisfy the pumping lemma. Analogously, we can prove that* $\mathcal{L}(L(N, M_0))$ *is not regular by choosing* $w = a^n b^n$, $x = a^{n-\delta}$, $y = a^\delta$, *with* $1 \leq \delta \leq n$, *and* $z = b^n$.

Note that if we modify the label of transition t_5 *as* $\mathcal{L}(t_5) = c$, *then this net is not diagnosable with respect to either definitions of diagnosability.* ■

In the next section, we develop necessary and sufficient conditions for diagnosability (both preceding definitions) of potentially unbounded Petri nets. We conclude this section by showing that Assumption (A3) is decidable. The decidability of the deadlock problem for general Petri nets has been proved by Cheng *et al.* in [9], where they also proved that the complexity of this problem is EXPSPACE-hard. We show how deadlock freeness after a *specific* transition is also decidable using a suitable net transformation.

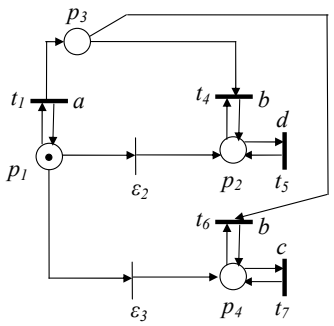

Figure 3: The Petri net system of Example 5.6.

Given a Petri net, we want that the only deadlock that is detected is a deadlock happening after the first fault transition fires. To do this, we duplicate each fault transition $t_{f,i}$ with a fault transition $t'_{f,i}$, namely the original and duplicate transitions have the same Pre and Post arcs as in the original net. Moreover, we add two new places p' and p'' and an unobservable transition ε' to the initial net. Place p' has a self-loop with ε', i.e., place p' has a Pre and Post arc with ε', and a Pre arc to each fault transition $t_{f,i}$; its initial marking is 1. Place p'' has a self loop with each fault transition $t'_{f,i}$ and a Post arc from each fault transition $t_{f,i}$; its initial marking is 0. This construction is illustrated in Fig. 4 for Petri net in Fig. 3. This transformation avoids the occurrence of a deadlock before any fault transition fires, since the additional transition ε' can always fire before a fault occurs. After the first fault transition occurs, transition ε' is disabled, the original fault transitions are disabled, but their duplicates are activated by means of the token put into place p''. Since we have duplicated all fault transitions, we are not modifying the net behavior.

The desired test for Assumption (A3) on the original Petri net now boils down to the standard deadlock freeness problem on the modified Petri net, which is decidable. Hence, this transformation shows that Assumption (A3) is decidable. We note however that at present, there are no known necessary and sufficient conditions based on structural analysis for deadlock freeness in general Petri nets. For special classes of nets, necessary and sufficient conditions based on structural properties of the net have been identified; see, e.g., [9, 10].

6 Analysis of Diagnosability

In this section we show how the diagnosability of an unbounded Petri net system can be checked by analyzing the CG of a special Petri net called *Verifier Net*. Note that the same approach can be applied to bounded Petri nets; in such a case the graph to be examined is the reachability graph of the verifier net.

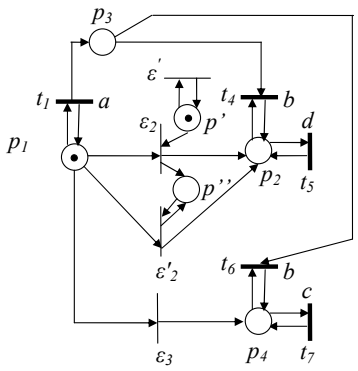

Figure 4: Net transformation for the verification of Assumption (A3) for the PN system of Example 5.6.

6.1 Verifier Net

For the sake of simplicity, and without loss of generality, we assume in the remainder of this paper that there is a single fault class; hence, the superscript i is omitted in T_f^i hereafter.

Let us consider the labeled Petri net system $\langle N, M_0, \mathcal{L}\rangle$, where $N = (P, T, Pre, Post)$, $T = T_o \cup T_u$, and $T_u = T_{reg} \cup T_f$. Let $\mathcal{L} : T \to L \cup \{\varepsilon\}$ be its labeling function. Let $N' = (P', T', Pre', Post')$ be its T'-induced subnet, where $T' = T \setminus T_f = T_o \cup T_{reg}$. Since we need to distinguish among places of N and N', we denote them as P and P', respectively, and assume that they are disjoint even if they represent the same places. Analogously, since we need to distinguish among the transitions of N and N', we denote them as T and T', respectively, and assume that they are disjoint even if they represent the same transitions. We assume that the Petri net system associated with N' is $\langle N', M_0', \mathcal{L}'\rangle$ where $M_0' = M_0$ and $\mathcal{L}'$ is equal to $\mathcal{L}$ restricted to T'.

The *Verifier Net* (denoted by VN herafter) system is the labeled Petri net system obtained by composing, in a manner made precise below, $\langle N', M_0', \mathcal{L}'\rangle$ with $\langle N, M_0, \mathcal{L}\rangle$ assuming that the synchronization is performed on the observable transition labels. This composition operation is related to parallel composition and to the construction of the verifier automaton of [43]. We denote it as $\langle \tilde{N}, \tilde{M}_0, \tilde{\mathcal{L}}\rangle$, where $\tilde{N} = (\tilde{P}, \tilde{T}, \tilde{Pre}, \tilde{Post})$, $\tilde{P} = P' \cup P$ and $\tilde{T} = \tilde{T}_o \cup (T'_{reg} \times \{\lambda\}) \cup (\{\lambda\} \times T_{reg}) \cup (\{\lambda\} \times T_f)$, where $\tilde{T}_o = \{(t', t) \mid t' \in T'_o, t \in T_o, \mathcal{L}'(t') = \mathcal{L}(t)\}$ and $\tilde{\mathcal{L}} : \tilde{T} \to (L \times L) \cup \{\varepsilon\}$.

The algorithm below shows how to construct the two matrices $\tilde{Pre}$ and $\tilde{Post}$.

Algorithm 6.1 ***Construction of the Verifier Net.***

Input: *Labeled Petri net system* $\langle N, M_0, \mathcal{L}\rangle$ *where* $N = (P, T, Pre, Post)$, $T = T_o \cup T_{reg} \cup T_f$ *and* $\mathcal{L} : T \to L \cup \{\varepsilon\}$.

Output: *VN labeled system* $\langle \tilde{N}, \tilde{M}_0, \tilde{\mathcal{L}}\rangle$, *where* $\tilde{N} = (\tilde{P}, \tilde{T}, \tilde{Pre}, \tilde{Post})$, *and* $\tilde{\mathcal{L}} : \tilde{T} \to (L \times L) \cup \{\varepsilon\}$.

1. *Let* $\langle N', M'_0, \mathcal{L}' \rangle$ *be the labeled Petri net system defined as discussed above.*
2. *Let* $\tilde{P} = P' \cup P$.
3. *Let* $\tilde{M}_0 = \begin{bmatrix} M'_0 \\ M_0 \end{bmatrix}$.
4. *For all transitions* $t_f \in T_f$,
   - *add a transition* $t \in \tilde{T}$ *denoted as* (λ, t_f);
   - *for all* $p \in P'$, *let* $\tilde{P}re(p,t) = \tilde{P}ost(p,t) = 0$;
   - *for all* $p \in P$, *let* $\tilde{P}re(p,t) = Pre(p,t_f)$ *and* $\tilde{P}ost(p,t) = Post(p,t_f)$.
5. *For all transitions* $t_{reg} \in T_{reg}$,
   - *add a transition* $t \in \tilde{T}$ *denoted as* (λ, t_{reg});
   - *for all* $p \in P'$, *let* $\tilde{P}re(p,t) = \tilde{P}ost(p,t) = 0$;
   - *for all* $p \in P$, *let* $\tilde{P}re(p,t) = Pre(p,t_{reg})$ *and* $\tilde{P}ost(p,t) = Post(p,t_{reg})$.
6. *For all transitions* $t'_{reg} \in T'_{reg}$,
   - *add a transition* $t \in \tilde{T}$ *denoted as* (t'_{reg}, λ);
   - *for all* $p \in P'$, *let* $\tilde{P}re(p,t) = Pre'(p,t'_{reg})$ *and* $\tilde{P}ost(p,t) = Post'(p,t'_{reg})$;
   - *for all* $p \in P$, *let* $\tilde{P}re(p,t) = \tilde{P}ost(p,t) = 0$.
7. *For all labels* $l \in L$,
   - *for any pair* t'_o, t_o *with* $t'_o \in T'_o$, $t_o \in T_o$, $\mathcal{L}'(t'_o) = \mathcal{L}(t_o) = l$,
     - *add a transition* $t \in \tilde{T}$ *denoted as* (t'_o, t_o);
     - *for all* $p \in P'$, *let* $\tilde{P}re(p,t) = Pre'(p,t'_o)$ *and* $\tilde{P}ost(p,t) = Post'(p,t'_o)$;
     - *for all* $p \in P$, *let* $\tilde{P}re(p,t) = Pre(p,t_o)$ *and* $\tilde{P}ost(p,t) = Post(p,t_o)$;
     - *label transition* t *with* (l,l). ■

The VN built using Algorithm 6.1 is a labeled Petri net system, where each transition is indicated by a pair, composed either by two transitions (in the case of observable transitions) or by one transition t and the symbol λ (in the case where t is an unobservable transition). No label, or equivalently the empty string ε, is associated with the unobservable transitions of the VN, while a label (l,l) is associated with the observable transitions. The set of places $\tilde{P}$ is the union of the set of places P of the Petri net system $\langle N, M_0, \mathcal{L} \rangle$, taken as input, and the set of places P' of the T'-induced subnet, where T' is the set of transitions obtained from T removing fault transitions in T_f (Step 2). The places in $\tilde{P}$ are initially marked as specified in M_0 and M'_0 (Step 3). All unobservable transitions, regular and faulty, indicated with the pair $(\lambda, \varepsilon_{uo})$, where $\varepsilon_{uo} \in T_{uo}$, are connected to places in P following the column of the Pre and $Post$ matrices relative to ε_{uo} (Steps 4 and 5). All unobservable transitions indicated with the pair $(\varepsilon'_{uo}, \lambda)$, where $\varepsilon'_{uo} \in T'_{uo}$, are connected to places in P' following the column of the Pre' and $Post'$ matrices relative to ε'_{uo}

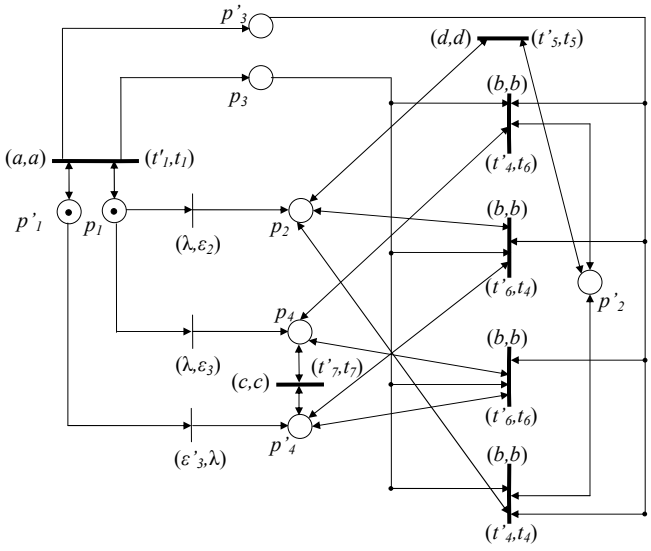

Figure 5: Verifier net $\langle \tilde{N}, \tilde{M}_0, \tilde{\mathcal{L}} \rangle$ where $\langle N, M_0, \mathcal{L} \rangle$ is the Petri net in Fig. 3.

(Step 6). Finally, observable transitions of the VN are indicated as (t'_o, t_o), where $t'_o \in T'_o, t_o \in T_o$ and $\mathcal{L}'(t'_o) = \mathcal{L}(t_o)$, and are connected to places P' following the column of the Pre' and $Post'$ matrices relative to t'_o and to places P following the column of the Pre and $Post$ matrices relative to t_o.

Example 6.2 *Figure 5 shows the VN of the Petri net system in Fig. 3, already introduced in Example 5.6. The set of places of the VN is obtained by the union of the set of places P of the Petri net system $\langle N, M_0 \rangle$ in Fig. 3 and the set of places P' of the T'-induced subnet. The T'-induced subnet is obtained from $\langle N, M_0 \rangle$ by removing fault transition ε_2; it is not drawn here.*

Observable transitions, denoted by black bars in Fig. 5, are indicated by two pairs (l, l) and (t'_o, t_o) (e.g., $(a, a,)$ (t'_1, t_1)), while unobservable transitions are indicated by only one pair (e.g., (λ, ε_3)), since no label is associated with them. Since label b is associated with two transitions (t_4 and t_6), the VN contains four transitions labeled (b, b).

Note that, to improve readability, if a place p has a self-loop with a transition t a double arrow arc is used in the figure (e.g., arc between p_1 and (t'_1, t_1)). ■

Proposition 6.3 *Given a labeled Petri net system $\langle N, M_0, \mathcal{L} \rangle$ and its VN, if a sequence $\tilde{\sigma} = (\gamma'_{i_1}, \gamma_{i_1})\,(\gamma'_{i_2}, \gamma_{i_2}) \ldots (\gamma'_{i_k}, \gamma_{i_k}) \in \tilde{T}^*$ is repetitive in the VN[2], then there exists a repetitive sequence $\sigma' = \gamma'_{i_1}\gamma'_{i_2} \ldots \gamma'_{i_k}$ in $\langle N', M'_0, \mathcal{L}' \rangle$ and a repetitive sequence $\sigma = \gamma_{i_1}\gamma_{i_2} \ldots \gamma_{i_k}$ in $\langle N, M_0, \mathcal{L} \rangle$ and both sequences σ and σ' have the same observable projection.*

Proof This result follows directly from the construction of VN. In fact, the two sequences have the same observable projection by construction of the VN. Moreover, the existence of a sequence $\tilde{\sigma} \in L(\tilde{N}, \tilde{M}_0)$ implies that $\sigma' \in L(N', M'_0)$ and $\sigma \in L(N, M_0)$. The firing sequences σ' and σ are

[2]Note that λ denotes the sequence of length zero, hence $\sigma'\lambda\sigma'' = \sigma'\sigma''$.

repetitive respectively in $\langle N', M'_0, \mathcal{L}' \rangle$ and in $\langle N, M_0, \mathcal{L} \rangle$, given that $\tilde{\sigma}$ is repetitive in the VN. □

6.2 Necessary and sufficient conditions for diagnosability

The following theorem shows how to determine the diagnosability of a Petri net system, starting from the reachability graph (bounded case) or coverability graph (unbounded case) of its VN; since we wish to treat these two cases simultaneously, we will write "reachability/coverability graph" hereafter, abbreviated as RG/CG. Let $F(VN)$ denote the set of *faulty nodes* in the RG/CG of the VN, namely the nodes that can be reached firing a path that contains a fault transition $t_f \in T_f$.

Theorem 6.4 *A labeled PN system $\langle N, M_0, \mathcal{L} \rangle$ satisfying assumptions A1 to A3 is* diagnosable *iff there does not exist any cycle associated with a firable repetitive sequence in the VN that is reachable starting from any node in the set $F(VN)$.*

Proof *We prove the* if *and* only if *statements separately.*

(Only if) *By contradiction, assume that in the RG/CG of the VN there exists a sequence of infinite length containing a fault, or equivalently, a cycle associated with a repetitive sequence that is firable for the VN starting from a node in $F(VN)$. From Propositions 4.2 and 6.3, this means that in the Petri net system $\langle N, M_0, \mathcal{L} \rangle$ there exist two firing sequences $s = \sigma_p(r)^q$ and $s' = \sigma'_p(r')^q$ with $q \in \mathbb{N}$, such that: σ_p contains a fault $t_f \in T_f$ but σ'_p does not, $\mathcal{L}(\sigma_p) = \mathcal{L}(\sigma'_p)$, r and r' are two repetitive sequences, and $\mathcal{L}(r) = \mathcal{L}(r')$. Thus there exist in $L(N, M_0)$ two sequences s and s', one containing a fault transition and the other one not containing it, both having the same observable projection, that can be made arbitrarily long using Definition 3.2. This violates the definition of diagnosability of $L(N, M_0)$ given in Definition 5.1, hence the Petri net is not diagnosable.*

(If) *We show that, under assumption A3, if the RG/CG of the VN does not contain a cycle associated with a repetitive sequence firable in the VN that is reachable starting from any node in the set $F(VN)$, namely there is no sequence of infinite length containing a fault, then the system is diagnosable. Let us consider what happens after an occurrence of a fault event in the system. By construction of the VN, the occurrence of a fault event will be captured by the RG/CG. In this case, if we consider two strings of events $s = \sigma_p$ and $s' = \sigma'_p$ such that s contains a fault transition $t_f \in T_f$ and s' does not, $\mathcal{L}(s) = \mathcal{L}(s')$, and attempt to extend these two strings in a manner that keeps their projections identical, the absence of a repetitive sequence in the VN after the said occurrence of a fault event will prevent this extension from growing arbitrarily long. Namely, we are unable to construct σ_1 and σ_2, as characterized in Definition 5.1. Since by assumption A3 the system does not enter a deadlock after a fault, this means that there is no violation of diagnosability.* □

The above result provides a necessary and sufficient condition for diagnosability. In Section 7, we will describe an implementable test that employs this necessary and sufficient condition.

Remark 6.5 *If the net system is bounded we just need to verify if starting from any node of the*

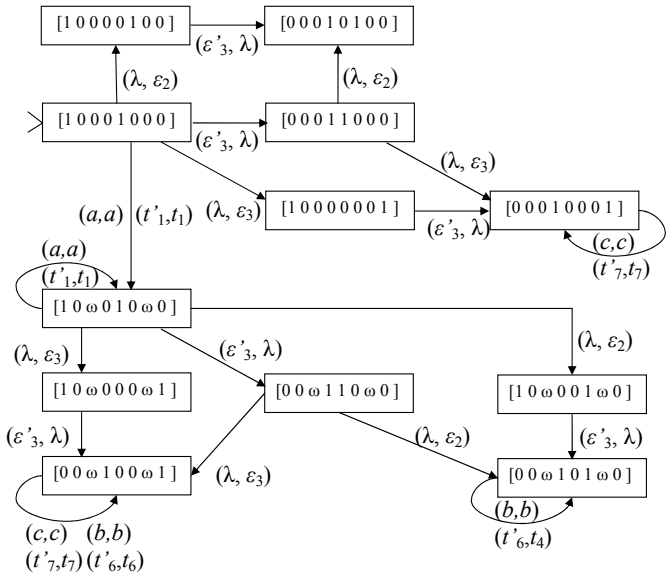

Figure 6: Coverability graph of the Verifier Net in Fig. 5.

reachability graph of its VN in the set $F(VN)$ there exists a cycle*; in such a case the system is not diagnosable. This is because the RG gives necessary and sufficient conditions for reachability [30] in the case of bounded nets; thus we are sure that the cycle is associated with a repetitive sequence. Therefore, in this case, the condition of Theorem 6.4 is easily implementable.* ■

Example 6.6 *Figure 6 shows the CG of the VN of Fig. 5. The Petri net system $\langle N, M_0, \mathcal{L}\rangle$ in Fig. 3 is diagnosable. In fact, looking at the CG of the VN, we observe that there is only one cycle ($[0\ 0\ \omega\ 1\ 0\ 1\ \omega\ 0]^T \xrightarrow{(b,b),(t'_6,t_4)} [0\ 0\ \omega\ 1\ 0\ 1\ \omega\ 0]^T$) that starts from a node in $F(VN)$, i.e., a node reached after firing the fault transition ε_2. However, this cycle is not associated with a repetitive sequence, since (t'_6, t_4) is not a repetitive sequence for the VN. To see this, let y_1 be the vector having all entries equal to zero except for the one associated to transition (t'_6, t_4); then we have that $\tilde{C} \cdot y_1 \not\geq \vec{0}$, where $\tilde{C} = \tilde{Post} - \tilde{Pre}$ is the incidence matrix of the VN.* ■

Finally, we note that the VN technique handles one fault class at a time. In the case of more than one fault class, we must build a VN for each fault class i, where all faults belonging to another fault class $j \neq i$ are considered as regular unobservable transitions.

6.3 Necessary and sufficient conditions for diagnosability in K steps

In this subsection we give necessary and sufficient conditions for diagnosability in K steps based on the RG/CG of the VN.

Theorem 6.7 *Let $\langle N, M_0, \mathcal{L}\rangle$ be a labeled Petri net system satisfying assumptions A1 to A3. There exists a finite K such that the system is diagnosable in K steps iff starting from any node of the RG/CG of its VN in the set $F(VN)$ there does not exist any cycle.*

Proof *In the case of bounded PNs, as discussed in Remark 6.5, the proof is straightforward.*

In the case of unbounded PNs, we prove the if *and* only if *statements separately.*

(Only if) *By contradiction, assume that in the CG of the VN there exists a node in $F(VN)$ from which a cycle is firable. This cycle can be associated either with a repetitive sequence or a non-repetitive sequence. If this cycle is associated with a repetitive sequence, then the system is not diagnosable by Theorem 6.4 and hence not diagnosable in K steps either, leading to a contradiction. If the cycle is associated with a non-repetitive sequence, then the system will be diagnosable (as proved in Theorem 6.4), thereby implying that the cycle cannot fire an infinite number of times. Moreover, the cycle must include ω-markings: if not the cycle is associated with a repetitive stationary sequence, a case we have excluded. The presence of ω-markings implies that there exists an increasing sequence from the initial state that can pump an indeterminate number of tokens in those places. Thus the sequence of the cycle will fire until it has consumed all tokens in those places. However, since the number of tokens pumped can be made arbitrarily large, we cannot fix a bound K, where K is the number of transitions that fire after a fault has occurred. Hence the Petri net is not diagnosable in K steps.*

(If) *We show that if the CG of the VN does not contain a node in $F(VN)$ from which a cycle is firable, then there exists a finite K such that the system is diagnosable in K steps. From Theorem 6.4, we know that the system is diagnosable because there is no cycle. Moreover, by construction of the VN and since there are no cycles, we can always determine after how many transitions the system will detect a fault. Thus, we can take K to be one more than the longest path after a fault occurs in the CG of the VN.* □

Since a bounded PN necessarily generates a regular language, by Proposition 5.3 and Remark 6.5, we can conclude that in the case of bounded PN systems, Theorem 6.7 is equivalent to Theorem 6.4.

Note that the methodology used in our approach for diagnosability analysis of bounded Petri nets is completely different from the one used in the classical automata approach [35].

6.4 Procedure to determine K in the case of diagnosability in K steps

In the previous subsection we presented a necessary and sufficient condition for diagnosability in K steps. We now present a procedure to directly compute the value of K for systems that are diagnosable in K steps.[3] This procedure avoids enumerating all the paths in the RG/CG of the VN in order to find the longest one after the firing of a fault transition, which was the argument used in the proof of Theorem 6.7. The desired value of K is directly read from the contents of a new place that is added to the net structure as described below.

First, we make a copy $\tilde{T}'$ of all transitions $\tilde{T}$ of the VN and connect them to the places of the VN in the same manner that the transitions $\tilde{T}$ are connected with the places in the VN. Then

[3]The method described in this subsection was suggested to the authors by Philippe Darondeau of IRISA Rennes (France). It is a pleasure to acknowledge his contribution.

we add to the VN three places: p_f, p'_f and p_K.
- Place p_f is initially marked with one token and has a self loop with each transition in $\tilde{T}$, except for fault transitions (λ, t_f), where $t_f \in T_f$; it also has a Pre arc with each fault transition (λ, t_f), where $t_f \in T_f$.
- Place p'_f is initially unmarked and has a self loop with each transition in $\tilde{T}'$ including fault transitions $(\lambda, t_f)'$, where $t_f \in T_f$; it also has a Post arc with each fault transition (λ, t_f), where $t_f \in T_f$.
- Place p_K is initially unmarked and has a Post arc from all transitions in $\tilde{T}'$ such as $(t'_o, t_o)'$, $(\lambda, t_{reg})'$, $(\lambda, t_f)'$.

Places p_f and p'_f do not alter the behavior of the net. As long as a fault transition (λ, t_f) does not fire, the transitions in $\tilde{T}$ are normally enabled. As soon as a fault transition (λ, t_f) fires, all transitions in $\tilde{T}$ are disabled, but all their copies, i.e., the transitions in $\tilde{T}'$, are enabled. Thus the language of the net is not modified. Place p_K is a counter and it allows us to take into account the number of transitions of the initial net that have fired after the firing of a fault transition (λ, t_f). Note that p_K is not taking into account transitions $(t'_{reg}, \lambda)'$ since we only wish to count transitions of the initial net N.

To determine the value of K for which the system is diagnosable in K steps, we build the RG/CG of this modified VN and we take K as one more than the maximum number of tokens contained in place p_K.

If there exist i different fault classes T_f^i we add j triples $(p_f^i, p_f'^i, p_K^i)$ in the VN. In such a case K will be the maximum value among all places p_K^j.

Finally, we note that if we apply the above procedure to a Petri net that is diagnosable but not diagnosable in K steps, the counter place p_K will for sure be ω-marked for some markings in the CG of the VN.

Example 6.8 *Let us consider the Petri net in Fig. 7, where* $T_o = \{t_1, t_2, t_3\}$ *and* $T_u = T_f = \{f\}$. *Let* $\mathcal{L}(t_1) = \mathcal{L}(t_2) = a$ *and* $\mathcal{L}(t_3) = b$. *We want to know if this net is diagnosable in* K *steps and in such a case we want to determine* K. *First, we built the VN of the net that is shown in Fig. 8, then we add the triple* (p_f, p'_f, p_K) *(just one triple since we only have one fault class), all copies of transitions* $\tilde{T}'$, *and we connect them as explained above. For the sake of clarity, we have drawn the place* p_f *and all its connections in blue, the place* p'_f *and all its connections in green, the place* p_K *and all its connections with dashed black lines, and the copies of transitions* $\tilde{T}'$ *and all their connections with the places of the VN in red.*

Looking at the CG shown in Fig. 10, where $p_9 = p_f$, $p_{10} = p'_f$ *and* $p_{11} = p_K$, *it is easy to see that the place* p_K *is bounded and its content is equal to* 1, *thus the Petri net is diagnosable in 2 steps. This means that we are able to detect that the fault has occurred after two transition firings in the worst case.* ■

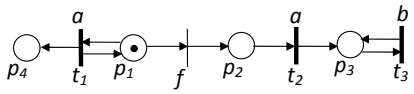

Figure 7: The Petri net system of Example 6.8.

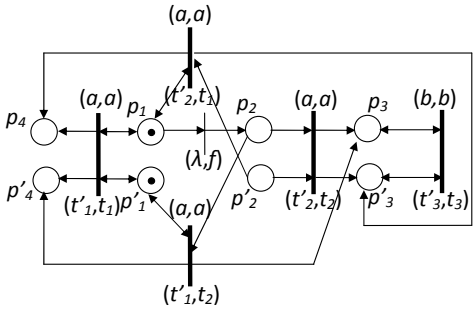

Figure 8: The VN of the Petri net system in Fig. 7.

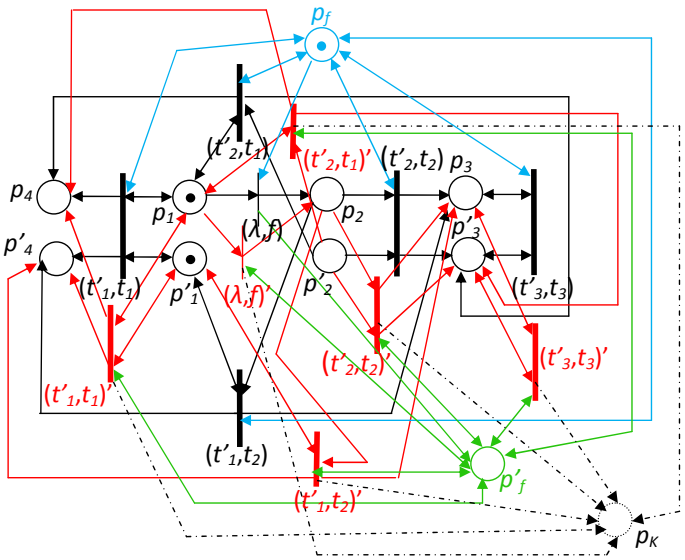

Figure 9: The VN of the Petri net system in Fig. 7 modified to compute the K of diagnosability in K steps.

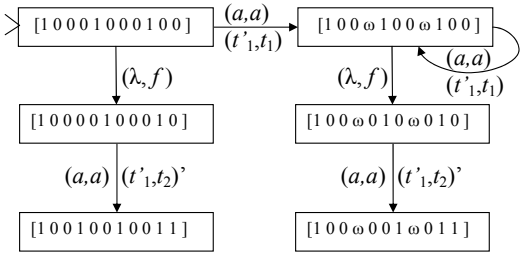

Figure 10: The CG of the Modified VN in Fig. 9.

7 Testing procedure for diagnosability

In this section we present computational procedures, for bounded and unbounded PNs, that implement the necessary and sufficient conditions for diagnosability of Section 6.2. We also discuss their computational complexity.

The construction of the VN is instrumental to the whole procedure. The VN has $2 \cdot m$ places and its number of transitions is of order $O(n^2)$, where m and n are respectively the number of places and transitions of the initial net N. Its construction is straightforward and we have developed a simple tool for this purpose. It requires to connect the transitions of the VN with its places as specified in Algorithm 6.1. Moreover, note that transitions and places are structural elements of a net. This means that a net can have a very large, even infinite, state space, even if its structure is very simple.

Let us discuss separately the case of bounded and unbounded PNs. As was mentioned in Remark 6.5, in the case of bounded net systems, once the VN has been built we simply need to explore its RG and look for cycles after the occurrence of a fault transition. In the case of bounded net systems, cycles always correspond to repetitive (stationary) sequences. We can first simplify the RG by erasing all nodes that do not belong to $F(VN)$, i.e., erasing all nodes that cannot be reached by a fault transition, and then examine if the resulting RG is acyclic. The total complexity of each of these two steps is linear in the sum of the number of states and transitions of the RG of the VN.

In the case of unbounded nets, the procedure is more complicated. Specifically, once the VN has been built we need to explore its CG and look for cycles, *not only elementary cycles*, associated with firable repetitive sequences, after the occurrence of a fault transition. To the best of our knowledge, the complexity of the construction of the CG is still an open issue. However, efficient tools are available to build the coverability graph, e.g., the Petri net tool TINA (Time Petri Nets Analyzer) [1]. To look for cycles associated with firable repetitive sequences (after the occurrence of a fault transition) in the CG of the VN we propose to use linear programming techniques. We describe our procedure in the remainder of this section.

We propose to use the VN to determine if a sequence is repetitive, and the components of a graph called Modified Coverability Graph (MCG), obtained starting from the CG of the VN, to identify cycles corresponding to sequences firable after the occurrence of a fault transition.

(1) We start with the CG of the VN and remove all the nodes that are not in $F(VN)$, i.e., all nodes that are not reachable by a path containing a fault transition; the graph obtained is called the Modified Coverability Graph (MCG) in the rest of this section. Then, we consider the maximal strongly connected components[4] of the MCG; assuming there are h such components, each of them will be denoted by an index α, with $\alpha \in \{1, \ldots, h\}$. The union of all these disconnected subgraphs necessarily contains all cycles of the MCG, i.e., all cycles in the original CG firable after a fault. Finally, for each component α of the MCG, we consider the corresponding *state*

[4] A directed graph is *strongly connected* if for each ordinate pair of nodes v, v' there exists an oriented path from v to v'.

machine labeled Petri net PN^α obtained as follows: to each node v_i corresponds a place p_i, and to each arc with transition label (t', t) directed from node v_i to v_j corresponds a transition (t', t) with a Pre arc from place p_i and a Post arc to place p_j. The cycles for net PN^α can be computed finding the firing vectors y^α that satisfy the equation $C^\alpha \cdot y^\alpha = 0$, where C^α is the incidence matrix of PN^α. If the solution y^α found is feasible then the cycle associated is firable by Proposition 4.2. Let $|T^\alpha|$ denote the cardinality of the set of transitions of PN^α.

(2) The desired repetitive sequences in the VN can be computed by finding the firing vectors y^{VN} that satisfy the equation $C^{VN} \cdot y^{VN} \geq 0$, where C^{VN} is the incidence matrix of the VN.

(3) Finally, to make sure that a cycle in component α corresponds to a repetitive sequence of VN, we map the two firing vectors. Specifically, we match the transitions (t'_{i_k}, t_{i_k}) of the VN with the corresponding transitions (t'_{i_k}, t_{i_k}) of PN^α. Note that to each transition in PN^α corresponds only one transition in the VN.

We can now state the main result of this section.

Theorem 7.1 *Let $\langle N, M_0, \mathcal{L}\rangle$ be a labeled Petri net system satisfying assumptions A1 to A3. Let VN be its verifier net and let CG and MCG be respectively the coverability graph and the modified coverability graph of the VN.*

If for each strongly connected component $\alpha \in \{1, \ldots, h\}$ of the MCG there exists no *feasible solution to the ILP problem*

$$\begin{cases} C^\alpha \cdot y^\alpha = 0 & (a) \\ C^{VN} \cdot y^{VN} \geq 0 & (b) \\ y^{VN} = B \cdot y^\alpha & (c) \\ y^{VN} \in \mathbb{N}^{|\tilde{T}|}, y^\alpha \in \mathbb{N}^{|T^\alpha|} & (d) \end{cases} \tag{4}$$

where

$$B(l, j) \;=\; \begin{cases} 1 & \text{if } (t'_{i_l}, t_{i_l})^{VN} = (t'_{i_j}, t_{i_j})^\alpha \\ 0 & \text{otherwise} \end{cases}$$

then the system is diagnosable.

Proof *It is sufficient to prove that if the assumption is verified, then starting from any node of the CG of the VN in $F(VN)$ there does not exist any cycle associated with a repetitive sequence in the VN. The result that the system is diagnosable then follows by an application of Theorem 6.4.*

If the ILP problem (4) *has no feasible solution for any of its strongly connected components, this means that there does not exist a cycle in any strongly connected component of the MCG (Constraints (a)) that can be associated with a firable repetitive sequence of the VN (Constraints (b) and (c)). Since, by construction, the MCG gives us necessary conditions for the reachability of a repetitive sequence after the firing of a fault transition, this means that there does not exist any cycle associated with a repetitive sequence starting from any node of the CG of its VN in $F(VN)$. Specifically, if there does not exist a firing vector y^α for net PN^α that satisfies the equation $C^\alpha \cdot y^\alpha = 0$, then by Fact 3.3 there does not exist a cycle σ^α in PN^α with $y^\alpha = \pi(\sigma^\alpha)$.*

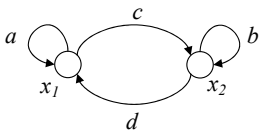

Figure 11: α-th strongly connected component of a Modified Coverability Graph

.

Moreover, if σ is a firing sequence of the VN, then, if it is repetitive, by Fact 3.3 its firing vector y^{VN} satisfies the equation $C^{VN} \cdot y^{VN} \geq 0$. □

Remark 7.2 *The sufficient condition in Theorem 7.1 is not necessary in general because it may happen that a solution for some strongly connected component α is found but this solution is not feasible because the subnet induced by the solution y^α is not connected. As an example, let us assume that the graph shown in Fig. 11 represents the α-th strongly connected component of the MCG. It may happen that ab is a repetitive sequence for the considered VN. In such a case the solution ab is found solving the ILP problem* (4)*. However, this solution is not feasible because it is a combination of two elementary cycles that are disconnected in the considered component.* ■

Summarizing, to test diagnosability we need to solve h ILP problems of the form given in (4). If no feasible solution is found, we can conclude that the system is diagnosable. On the contrary, if there are solutions, for each solution y^α, we need to verify its feasibility, namely, if it corresponds to a cycle in the component. As soon as we find one feasible solution we can state that the system is *not* diagnosable.

We present an illustrative example of the above testing procedure for unbounded nets.

Example 7.3 *Let us consider the Petri net in Fig. 3 introduced in Example 5.6. The MCG of this net is shown in Fig. 12. The arrows in the figure indicate the nodes in $F(VN)$, i.e., the nodes that are reached firing (or after the firing of) the fault transition ε_2. The only strongly connected component of the MCG containing cycles is the one composed of the self loop at the node marked $M = [0\ 0\ \omega\ 1\ 0\ 1\ \omega\ 0]^T$; the corresponding state machine PN^1 is shown in Fig. 13. We solve the ILP problem in* (4) *corresponding to that strongly connected component. Looking at the VN in Fig. 5 and the PN^1 in Fig. 13, we write constraints (c) $y^{VN} = B \cdot y^1$, where $B(l, j) = 1$ for $(t'_{i_l}, t_{i_l})^{VN} = (t'_6, t_4)$ and $(t'_{i_j}, t_{i_j})^1 = t_3^{PN^1}$. We find that no solution exists; thus the net is diagnosable, as stated in Example 6.6.* ■

We conclude this section by showing how the diagnosability of the net can be determined by solving a set of h *Linear Programming* (LP) problems, instead of solving the set of h ILP problems of the form (4), where h is the number of strongly connected components of the MCG. This results in considerable savings in terms of computational complexity.

In [7] we define a special class of linear constraint sets (CSs).

Definition 7.4 *[7] Given $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, consider the linear constraint set:*

$$\mathcal{C}(A, b) = \{x \in \mathbb{R}^n \mid Ax \geq b\}.$$

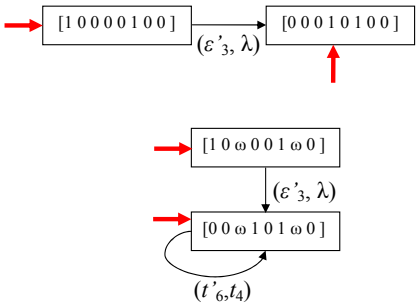

Figure 12: Modified Coverability Graph of the Verifier Net in Fig. 5.

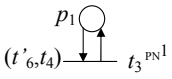

Figure 13: Unique strongly connected component of the MCG in Fig. 12.

The set $\mathcal{C}(A,b)$ is called:

- ideal *if $x \in \mathcal{C}(A,b)$ implies $\alpha x \in \mathcal{C}(A,b)$ for all $\alpha \geq 1$;*
- rational *if $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$, i.e., if the entries of matrix A and vector b are rational.*

■

We cite two results from [7] that provide a simple characterization of ideal CSs.

Proposition 7.5 *[7] A linear constraint set $\mathcal{C}(A,b)$ is ideal if $b \geq 0$.*

Proposition 7.6 *[7] If a CS is ideal and rational, then it has a feasible solution if and only if it has a feasible integer solution.*

It is straightforward to show that the CS (4) is ideal and rational; observe that it can be rewritten as:

$$\begin{cases} C^{\alpha} \cdot y^{\alpha} \geq 0 & (a1) \\ -C^{\alpha} \cdot y^{\alpha} \geq 0 & (a2) \\ C^{VN} \cdot y^{VN} \geq 0 & (b) \\ y^{VN} - B \cdot y^{\alpha} \geq 0 & (c1) \\ -y^{VN} + B \cdot y^{\alpha} \geq 0 & (c2) \\ y^{VN}, y^{\alpha} \geq 0 & (d) \end{cases} \tag{5}$$

In view of Proposition 7.6, we conclude that we can obtain a sufficient condition for the diagnosability properties of an unbounded Petri net system by solving h LP problems of the form (5), where h is the number of strongly connected components of the MCG. The number of constraints in (5) can be upper bounded. In particular, the number of Constraints (a1) and (a2) is equal

to the cardinality of the set of places of PN^{α}. The number of Constraints (b) is equal to $2 \cdot m$, where m is the number of places of the original net N. Finally, the number of Constraints (c1) and (c2) is equal to the cardinality of the set of transitions of the VN ($|\tilde{T}|$).

8 Online diagnosis

The results presented in this paper solve the problem of determining *diagnosability* of bounded and unbounded PNs, according to the notions of diagnosability formulated in Section 5. However, the constructs upon which the necessary and sufficient conditions of diagnosability are based, namely, the VN and its RG/CG, cannot be used as is for online diagnosis, as they include unobservable transitions. One could try to determinize the CG of the original net for on-line diagnosis, an approach related to the construction of diagnoser automata for a system modeled by a finite-state automaton, but this approach would suffer from the fact that for unbounded PNs, the CG only gives necessary conditions for reachability. Thus, there could be indeterminate cycles in the CG, i.e., cycles that would normally lead to non-diagnosability of the system, but these cycles may be reached by firing sequences that are *not* firable in the considered net system, thereby not leading to a violation of diagnosability.

We already proposed solutions to the online diagnosis of labeled Petri nets in [14], [6]; they are briefly recalled in the following discussion. Note that we presented these methods for bounded Petri nets, but they could potentially be generalized to deal with unbounded Petri nets.

The approach in [14] requires an exhaustive enumeration of the set of all possible reachable markings each time an event is observed. A vector of cardinality r, where r is the number of fault classes, is associated to each possible reachable marking M. The i-th entry of this vector is equal to 1 if in reaching M, one or more fault transitions belonging to the i-th fault class have occurred; the entry is set to 0 otherwise. Online diagnosis is performed by examining the components of all such vectors after each observable event. In [6] we give a method to perform online diagnosis using the notions of *basis marking* and *justification*. Given an observed string w, a basis marking M_b is a marking that is reached firing w and all those unobservable transitions strictly necessary to enable w. A justification is the minimal firing sequence of unobservable transitions that, interleaved with w, enables its firing. The notion of basis marking allows us to reduce the reachability space; in fact, each time an observable transition fires we do not have to enumerate all the markings consistent with the observation but only a subset of them. Each time an observable transition fires, a diagnosis state is computed based on the set of pairs (reached basis marking, corresponding justification).

9 Conclusion

We have presented the first set of necessary and sufficient conditions for diagnosability and diagnosability in K steps of possibly unbounded labeled Petri nets. Our approach is based on the

new concept of verifier net, and on the exploration of its reachability or coverability (unbounded case) graph for the existence of repetitive sequences. We have also presented new results that provide a connection between the above two notions of diagnosability in the case of Petri nets generating regular languages of transition firings. Moreover, we have presented a method to compute the bound K in the case of systems that are diagnosable in K steps. Finally, we have proposed a computational procedure to test the necessary and sufficient conditions for unbounded Petri nets, based on the solution of a number of linear programming problems. Future works of interest include the development of new methods for online diagnosis of unbounded Petri nets and the study of alternative methods, that do not require the construction of the reachability graph of the VN, but that exploit the structure of the VN to verify the diagnosability of bounded Petri nets.

References

[1] TINA website: http://homepages.laas.fr/bernard/tina.

[2] F. Basile, P. Chiacchio, and G. De Tommasi. An efficient approach for online diagnosis of discrete event systems. *IEEE Trans. on Automatic Control*, 54(4), 2009.

[3] A. Benveniste, E. Fabre, S. Haar, and C. Jard. Diagnosis of asynchronous discrete event systems: A net unfolding approach. *IEEE Trans. on Automatic Control*, 48(5):714–727, May 2003.

[4] R.K. Boel and J.H. van Schuppen. Decentralized failure diagnosis for discrete-event systems with costly communication between diagnosers. In *Proc. WODES'02: 6th Work. on Discrete Event Systems*, pages 175–181, October 2002.

[5] M.P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu. Diagnosability analysis of unbounded Petri nets. In *Proc. 48th IEEE Conf. on Decision and Control*, Shanghai, China, dec 2009.

[6] M.P. Cabasino, A. Giua, M. Pocci, and C. Seatzu. Discrete event diagnosis using labeled Petri nets. An application to manufacturing systems. *Control Engineering Practice*, 19(9):989–1001, September 2011.

[7] M.P. Cabasino, A. Giua, and C. Seatzu. Linear programming techniques for the identification of place/transition nets. In *47th IEEE Conf. on Decision and Control*, Cancun, Mexico, 2008.

[8] M.P. Cabasino, A. Giua, and C. Seatzu. Diagnosability of bounded Petri nets. In *Proc. 48th IEEE Conf. on Decision and Control*, Shanghai, China, dec 2009.

[9] A. Cheng, J. Esparza, and J. Palsberg. Complexity results for 1-safe nets. *Theor. Comput. Sci.*, 147(1-2), 1995.

[10] F. Chu and X.-L. Xie. Deadlock analysis of Petri nets using siphons and mathematical programming. *IEEE Trans. on Robot Autom.*, 13(6):793–804, 1997.

[11] S.L. Chung. Diagnosing pn-based models with partial observable transitions. *International Journal of Computer Integrated Manufacturing*, 12 (2):158–169, 2005.

[12] R. Debouk, S. Lafortune, and D. Teneketzis. Coordinated decentralized protocols for failure diagnosis of discrete-event systems. *Discrete Events Dynamical Systems*, 10(1):33–86, January 2000.

[13] M. Dotoli, M.P. Fanti, A.M. Mangini, and W. Ukovich. Fault detection of discrete event systems using Petri nets and integer linear programming. *Automatica*, 45:2665–2672, 2009.

[14] S. Genc and S. Lafortune. Distributed diagnosis of discrete event systems using Petri nets. In *Proc. of the 24th ATPN*, pages 316–336, June 2003.

[15] S. Genc and S. Lafortune. Distributed diagnosis of place-bordered Petri nets. *IEEE Trans. on Automation Science and Engineering*, 4(2):206–219, 2007.

[16] Stefan Haar. Qualitative diagnosability of labeled Petri nets revisited. In *Proc. 48th IEEE Conf. on Decision and Control*, pages 1248–1253.

[17] C.N. Hadjicostis and G.C. Veghese. Monitoring discrete event systems using Petri net embeddings. *Lecture Notes in Computer Science*, 1639:188–207, 1999.

[18] S. Hashtrudi Zad, R.H. Kwong, and W.M. Wonham. Fault diagnosis in discrete-event systems: framework and model reduction. *IEEE Trans. on Automatic Control*, 48(7):1199–1212, July 2003.

[19] J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to Automata Theory, Languages and Computation (Third Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.

[20] M. Jantzen. Complexity of place/transition nets. In *Advances in Petri nets 1986, part I on Petri nets: central models and their properties*, 1987.

[21] S. Jiang and R. Kumar. Failure diagnosis of discrete-event systems with linear-time temporal logic specifications. *IEEE Trans. on Automatic Control*, 49(6):934–945, June 2004.

[22] S. Jiang, R. Kumar, and H. E. Garcia. Diagnosis of repeated/intermittent failures in discrete event systems. *IEEE Transactions on Robotics and Automation*, 19(2):310–323, 2003.

[23] G. Jiroveanu and R.K. Boel. The diagnosability of Petri net models using minimal explanations. *IEEE Trans. on Automatic Control*, 55(7):1663–1668, 2010.

[24] R.M. Karp and R.E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, (2):147–195, May 1969.

[25] D. Lefebvre and C. Delherm. Diagnosis of DES with Petri net models. *IEEE Trans. on Automation Science and Engineering*, 4(1):114–118, 2007.

[26] F. Lin. Diagnosability of discrete event systems and its applications. *Discrete Event Dynamic Systems*, 4(2):197–212, 1994.

[27] F. Lin, J. Markee, and B. Rado. Design and test of mixed signal circuits: a discrete event approach. In *Proc. 32rd IEEE Conf. on Decision and Control*, pages 246–251, 1993.

[28] J. Lunze and J. Schroder. Sensor and actuator fault diagnosis of systems with discrete inputs and outputs. 34(3):1096–1107, April 2004.

[29] A. Madalinski, F. Nouioua, and P. Dague. Diagnosability verification with Petri net unfoldings. *International Journal of Knowledge-Based and Intelligent Engineering Systems*, 14(2):49–55, 2010.

[30] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.

[31] J. Prock. A new tecnique for fault detection using Petri nets. *Automatica*, 27(2):239–245, 1991.

[32] A. Ramirez-Treviño, E. Ruiz-Beltràn, I. Rivera-Rangel, and E. Lopez-Mellado. Online fault diagnosis of discrete event systems. A Petri net-based approach. *IEEE Trans. on Automation Science and Engineering*, 4(1):31–39, 2007.

[33] Y. Ru and C.N. Hadjicostis. Fault diagnosis in discrete event systems modeled by partially observed Petri nets. *Discrete Event Dynamic Systems*, 19(4):551–575, 2009.

[34] M. Sampath, S. Lafortune, and D. Teneketzis. Active diagnosis of discrete-event systems. *IEEE Trans. on Automatic Control*, 43(7):908–929, July 1998.

[35] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Diagnosability of discrete-event systems. *IEEE Trans. on Automatic Control*, 40(9):1555–1575, September 1995.

[36] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Failure diagnosis using discrete-event models. *IEEE Trans. Control Systems Technology*, 4(2):105–124, 1996.

[37] V.S. Sreenivas and M.A. Jafari. Fault detection and monitoring using time Petri nets. *IEEE Trans. Systems, Man and Cybernetics*, 23(4):1155–1162, 1993.

[38] T. Ushio, L. Onishi, and K. Okuda. Fault detection based on Petri net models with faulty behaviors. In *Proc. SMC'98: IEEE Int. Conf. on Systems, Man, and Cybernetics (San Diego, CA, USA)*, pages 113–118, October 1998.

[39] R. Valk and G. Vidal-Naquet. Petri nets and regular languages. *Journal of Computer and System Sciences*, 23(3):299–325, 1981.

[40] Y. Wen and M. Jeng. Diagnosability analysis based on T-invariants of Petri nets. In *Networking, Sensing and Control, 2005. Proceedings, 2005 IEEE.*, pages 371– 376, March 2005.

[41] Y. Wen, C. Li, and M. Jeng. A polynomial algorithm for checking diagnosability of Petri nets. In *Proc. SMC'05: IEEE Int. Conf. on Systems, Man, and Cybernetics*, pages 2542–2547, October 2005.

[42] Y. Wu and C.N. Hadjicostis. Algebraic approaches for fault identification in discrete-event systems. *IEEE Trans. Robotics and Automation*, 50(12):2048–2053, 2005.

[43] T.-S. Yoo and S. Lafortune. Polynomial-time verification of diagnosability of partially observed discrete-event systems. *IEEE Trans. on Automatic Control*, 47(9):1491–1495, September 2002.

[44] T.S. Yoo and H.E. Garcia. Event diagnosis of discrete-event systems with uniformly and nonuniformly bounded diagnosis delays. *Discrete Events Dynamical Systems*, 19(2):167–187, 2009.