Codiagnosability analysis of bounded Petri nets

Ning Ran, Hongye Su, Alessandro Giua, and Carla Seatzu

***Abstract*—In this paper we propose a novel approach to perform codiagnosability analysis of labeled bounded Petri nets. A set of sites observe the system evolution, each one with its own observation mask. Sites do not exchange information with each other but communicate with a coordinator. The coordinator is able to detect a fault if and only if at least one site is able to do that. In a previous work by some of us it has been proven that a necessary and sufficient condition for codiagnosability under such a framework, is the absence of sequences that are "ambiguous" with respect to all sites and whose length may grow indefinitely after the occurrence of some fault. The novelties of the approach consist in using the notion of basis markings to avoid exhaustive enumeration of the set of reachable markings, and in the construction of an automaton, called Verifier, that allows one to detect the presence of ambiguous sequences.**

Finally, we introduce the notion of K-codiagnosability: a system is K-codiagnosable if and only if faults can be detected in the above framework within at most K observations after their occurrence. An algorithm is provided to compute the smallest value of K such that the system is K-codiagnosable.

***Index Terms*—Discrete event systems, Petri nets, fault diagnosis, Verifier.**

I. Introduction

Solving a diagnosability problem consists in determining if, once a fault has occurred, its occurrence can be detected in a finite number of steps. This problem has been extensively investigated in a centralized setting [1]–[7]. However, nowadays systems are often intrinsically distributed. This is the reason why in recent years, a series of decentralized approaches have been developed both in the automata and Petri net framework [8]–[14].

The notion of codiagnosability has been first introduced by Qiu and Kumar in [9] under the assumption that all local diagnosers do not communicate with each other and only send information to a coordinator. Algorithms with polynomial complexity in the size of the automaton modeling the plant and the nonfaulty specification are provided for verifying codiagnosability and computing the bound in the delay of diagnosis. Yin and Lafortune [10] investigate the transformation from codiagnosability to coobservability under dynamic observations, and present an approach for the verification of codiagnosability. Their results complement those in Wang *et al.* [11] who study the reverse transformation, namely from coobservability to codiagnosability, thereby resulting in a thorough characterization of the relationship between the two notions of codiagnosability and coobservability and their verification. Takai and Ushio [12] consider the decentralized failure diagnosis problem of Mealy automata with nondeterministic output functions. They introduce an extended version of codiagnosability as a condition for the existence of a decentralized diagnoser, and give an algorithm to verify codiagnosability.

In recent years, some results have been presented for decentralized fault diagnosis of bounded Petri nets. Cabasino *et al.* [14] present a procedure to analyze the diagnosability of a Petri net system in a decentralized framework. Inspired by Debouk *et al.* [8], they first prove that the absence of failure ambiguous sequences is a necessary and sufficient condition for codiagnosability, and give a procedure to verify the absence of such kind of sequences in bounded and unbounded Petri net systems. The verification is based on the analysis of the reachability/coverability graph of a particular Petri net called Modified Verifier Net, which is an extension of the Verifier Net introduced in [4] to analyze diagnosability in a centralized setting. However, the complexity of this approach may increase exponentially with the size of the net (structure and number of tokens in the initial marking) thus making it unfeasible in practical situations. In [13] Basile *et al.* propose an approach for the analysis of K-codiagnosability that differs from most of the other approaches in the literature (including ours) for the fact that it is not based on the off-line construction of an automaton, e.g., a Verifier, but is based on the solution of an integer linear programming problem. Even if the approach by Basile *et al.* is NP-complete being it based on integer linear programming, it could be (as the authors claim) a promising approach to formalize the problem of optimally select sensors to be attached to transitions to guarantee K-codiagnosability. Furthermore, the approach proposed by Basile *et al.* does not require that the unobservable-induced subnets with respect to local sites are acyclic.

In this paper, we propose an approach for codiagnosability analysis that is still based on the idea of constructing a Verifier, but, thanks to the notion of basis makings [15] we avoid an exhaustive enumeration of the state space. As in [14], we analyze codiagnosability by looking at failure ambiguous sequences. We define an automaton called Verifier that enables us to check the existence of special cycles called F-cycles, which correspond to failure ambiguous sequences having infinite length after the fault. In this way, the problem of codiagnosability analysis reduces to the problem of looking for F-cycles in the Verifier. Furthermore, we introduce the definition of K*-codiagnosability*, which is a generalization of K-diagnosability. In simple words, a labeled Petri net system

This work is supported by National Natural Science Foundation of China under Grant 61134007, National Basic Research Program of China under Grant 2013CB035406, the 111 Project under Grant B07031.

N. Ran is with College of Electronic and Information Engineering, Hebei University, Baoding 071002, China. ranning87@hotmail.com

H. Su is with State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China. hysu@iipc.zju.edu.cn

A. Giua is with Aix Marseille Univ, Université de Toulon, CNRS, ENSAM, LSIS, Marseille, France and also with DIEE, University of Cagliari, Cagliari 09124, Italy. alessandro.giua@lsis.org, giua@diee.unica.it

C. Seatzu is with DIEE, University of Cagliari, Cagliari 09124, Italy. seatzu@diee.unica.it

monitored by a set of local sites, is K-codiagnosable with respect to a given fault class if faults in that class can be detected in at most K observations after the occurrence of the fault. An algorithm to compute the smallest value of K such that a system is K-codiagnosable is provided.

Note that this paper is an extended version of [16]. Novelties are: (1) a more exhaustive survey of the literature; (2) a slightly different definition of failure ambiguous string that does not assume that the system is diagnosable in a centralized way; (3) formal proofs of all the results; (4) the definition of K-codiagnosability and an algorithm for its analysis.

II. Background on labeled Petri nets

A Petri net (PN) [17] is a 4-tuple $N = (P, T, F, W)$, where P and T are finite, non-empty, and disjoint sets, $F \subseteq (P \times T) \cup (T \times P)$ is called the flow relation of the net, W is a mapping that assigns a non negative integer weight to an arc: $W(x, y) > 0$ iff $(x, y) \in F$, and $W(x, y) = 0$ otherwise, where $x, y \in P \cup T$. The incidence matrix $[N]$ of N is a $|P| \times |T|$ integer matrix with $[N](p, t) = W(t, p) - W(p, t)$. Let $x \in P \cup T$ be a node of net N. The preset of x is defined as ${}^\bullet x = \{y \in P \cup T \mid (y, x) \in F\}$ while the postset of x is defined as $x^\bullet = \{y \in P \cup T \mid (x, y) \in F\}$.

A marking m of a PN N is a mapping from P to $\mathbb{N} = 0, 1, 2, ...$: $m(p)$ denotes the number of tokens in place p. (N, m_0) denotes a PN system with an initial marking m_0.

A transition t is enabled at a marking m if $\forall p \in {}^\bullet t, m(p) \geq W(p, t)$. This fact is denoted by $m[t\rangle$ while $m[\sigma\rangle$ is used to denote that the transition sequence $\sigma = t_1 t_2 ... t_k$ is enabled at m. We denote by $|\sigma|$ the length of the sequence σ. The Parikh vector of σ is denoted by $\pi(\sigma)$. The set of all 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 write $t \in \sigma$ to denote that a transition t is contained in σ, $T' \cap \sigma \neq \emptyset$ to denote that there is at least one transition in T' contained in σ and $T' \cap \sigma = \emptyset$ to denote that there is no transition in T' contained in σ, where T' is a set of transitions.

Firing t yields a new marking m' such that $\forall p \in P, m'(p) = m(p) + [N](p, t)$, which is denoted by $m[t\rangle m'$. Marking m'' is said to be reachable from m if there exists a transition sequence σ such that $m[\sigma\rangle m''$. The set of markings reachable from m in N is called the reachability set of (N, m) and is denoted by $R(N, m)$.

A PN is said to be bounded if there exists a positive constant k such that $\forall p \in P$, $\forall m \in R(N, m_0)$, $m(p) \leq k$. It is unbounded if it is not bounded.

Given a PN system (N, m_0), $t \in T$ is live under m_0 if $\forall m \in R(N, m_0)$, $\exists m' \in R(N, m)$, $m'[t\rangle$. A PN system (N, m_0) is: *live* if $\forall t \in T$, t is live under m_0; *dead* under m_0 if $\nexists t \in T$, $m_0[t\rangle$; *deadlock-free* if $\forall m \in R(N, m_0)$, $\exists t \in T$, $m[t\rangle$.

Given a PN $N = (P, T, F, W)$ and a set $T' \subseteq T$ of transitions, we define T'-induced subnet of N the new PN $N' = (P, T', F', W)$, where F' is the restriction of F to $(P \times T') \cup (T' \times P)$. The net N' can be obtained from N by removing all transitions in $T \setminus T'$.

A Petri net with no directed circuits is said to be acyclic.

A labeled PN system is a triple $(N, m_0, \mathcal{L})$, where (N, m_0) is a PN system, $\mathcal{L}$ is a labeling function $\mathcal{L} : T \to A \cup \{\varepsilon\}$ that assigns to each transition in T either a symbol from a given alphabet A or the empty sequence ε.

We use T_u to denote the set of transitions whose labels are ε, and T_o to denote the set of transitions whose labels are the symbols in A. T_u and T_o are called the set of unobservable and observable transitions, respectively. $[N]_u$ (or $[N]_o$) is used to denote the restriction of the incidence matrix $[N]$ to T_u (or T_o). Given $\sigma \in T^*$, we denote $P_u(\sigma)$ (or $P_o(\sigma)$) the projection of σ over T_u (or T_o).

The labeling function can be naturally extended to define the projection operator $\mathcal{L} : T^* \to A^*$ as follows: (1) $\mathcal{L}(\varepsilon) = \varepsilon$; (2) $\mathcal{L}(t) = l$ for some $l \in A$, if $t \in T_o$; (3) $\mathcal{L}(t) = \varepsilon$, if $t \in T_u$; and (4) $\mathcal{L}(\sigma t) = \mathcal{L}(\sigma)\mathcal{L}(t)$, if $\sigma \in T^* \wedge t \in T$.

Moreover, $\mathcal{L}^{-1}(w)$ is used to denote the set of all transition sequences consistent with $w \in A^*$, i.e., $\mathcal{L}^{-1}(w) = \{\sigma \in L(N, m_0) \mid \mathcal{L}(\sigma) = w\}$. Using the extended labeling function, the language of transition labels is therefore denoted by $\mathcal{L}(L(N, m_0))$.

Let $K \subseteq T^*$ be a language, we use K/σ to denote the post-language of K after σ, i.e., $K/\sigma = \{\sigma' \in T^* \mid \sigma\sigma' \in K\}$.

III. Problem statement

The unobservable transition set is partitioned as $T_u = T_f \cup T_{reg}$, where T_f is the set of fault transitions and T_{reg} is the set of unobservable but regular transitions. We use $[N]_{reg}$ to denote the restriction of the incidence matrix to T_{reg}. The fault transition set T_f is partitioned into r different subsets T_f^i that model different fault classes, where $i = 1, 2, ..., r$.

The PN is monitored by a set $\mathcal{J} = \{1, 2, ..., \nu\}$ of sites. Each site knows the structure of the net and observes the evolution of the system by its own mask. Sites may send information to a coordinator but do not communicate with the other sites. We assume that the coordinator follows protocol 3 in [8], i.e., a fault in a given class is diagnosed if and only if at least one local site detects its occurrence.

The set of transitions that are observable (resp., unobservable) for site $j \in \mathcal{J}$ is denoted by $T_{o,j} \subseteq T_o$ (resp., $T_{u,j} \subseteq T$). The alphabet of the j-th site is denoted $A_j \subseteq A$, and the labeling function associated with the j-th site is $\mathcal{L}_j(t) = \mathcal{L}(t)$ if $\mathcal{L}(t) \in A_j$, otherwise it is $\mathcal{L}_j(t) = \varepsilon$.

Given a transition sequence $\sigma \in L(N, m_0)$, $w_j = \mathcal{L}_j(\sigma)$ is used to denote the sequence of labels in A_j associated with σ observed by the j-th site.

We make the following assumptions:

A1) The PN system is deadlock-free after the occurrence of any fault;

A2) The PN system is bounded;

A3) The $T_{u,j}$-induced subnets are acyclic for all $j = 1, 2, ..., \nu$.

For $i = 1, \ldots, r$, we use $\Psi(T_f^i)$ to denote the set of all sequences in $L(N, m_0)$ that end with a transition in T_f^i.

Definition 1: Let $(N, m_0, \mathcal{L})$ be a labeled PN system that is deadlock-free after the occurrence of any fault $t_f \in T_f$. Assume that $(N, m_0, \mathcal{L})$ is monitored by a set $\mathcal{J} = \{1, 2, ..., \nu\}$ of local sites. The labeled PN system $(N, m_0, \mathcal{L})$ is *codiagnosable* wrt the i-th fault class T_f^i if

$$(\forall s \in \Psi(T_f^i)),\ (\exists K \in \mathbb{N}),\ (\forall \sigma \in L(N, m_0)/s),\ |\sigma| \geq K \Rightarrow$$

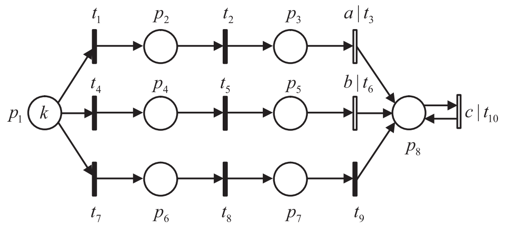

Fig. 1: A Labeled PN system $(N, m_0, \mathcal{L})$.

$$(\exists j \in \mathcal{J}),\ (\forall \sigma' \in \mathcal{L}_j^{-1}(\mathcal{L}_j(s\sigma))),\ T_f^i \cap \sigma' \neq \emptyset.$$

The labeled PN system $(N, m_0, \mathcal{L})$ is *codiagnosable* if it is codiagnosable wrt all fault classes.

In simple words, $(N, m_0, \mathcal{L})$ is codiagnosable wrt T_f^i if, once a fault in T_f^i has occurred, there exists at least one site that detects it within a finite delay. In this paper we provide an approach to analyze codiagnosability that is based on the notion of failure ambiguous string.

Definition 2: Consider a labeled PN system $(N, m_0, \mathcal{L})$ whose labeling function $\mathcal{L}$ is defined over an alphabet A. Assume that $(N, m_0, \mathcal{L})$ is monitored by a set $\mathcal{J} = \{1, 2, ..., \nu\}$ of local sites. A sequence $\sigma \in T^*$ such that $T_f^i \cap \sigma \neq \emptyset$ is said to be *failure ambiguous* wrt T_f^i if there exist ν sequences $\sigma_1, \sigma_2, ..., \sigma_\nu \in T^*$, not necessarily distinct, such that

$T_f^i \cap \sigma_j = \emptyset$ and $\mathcal{L}_j(\sigma) = \mathcal{L}_j(\sigma_j)$, $j = 1, 2, ..., \nu$.

In words, a sequence σ containing some fault transitions in T_f^i is failure ambiguous wrt T_f^i if σ is ambiguous for all sites.

A very similar definition has been proposed in [14]. However, in that case it was assumed that the sequence that is ambiguous for all sites is not ambiguous for a centralized diagnoser who observes all the labels observed by the local sites.

Example 1: Consider the labeled PN system $(N, m_0, \mathcal{L})$ in Fig. 1, where $T_o = \{t_3, t_6, t_{10}\}$, $T_u = \{t_1, t_2, t_4, t_5, t_7 - t_9\}$, $T_f = \{t_9\}$, $m_0 = [k\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T$, and k is a positive constant. The labeling function is defined as follows: $\mathcal{L}(t_3) = a$, $\mathcal{L}(t_6) = b$ and $\mathcal{L}(t_{10}) = c$. Assume that the PN is monitored by two local sites whose alphabets are equal to $A_1 = \{a, c\}$ and $A_2 = \{b, c\}$, respectively. The faulty sequence $\sigma = t_7t_8t_9t_{10}$ is failure ambiguous wrt T_f. In fact, there exist two fault-free sequences $\sigma_1 = t_4t_5t_6t_{10}$ and $\sigma_2 = t_1t_2t_3t_{10}$ such that $\mathcal{L}_1(\sigma) = \mathcal{L}_1(\sigma_1) = c$ and $\mathcal{L}_2(\sigma) = \mathcal{L}_2(\sigma_2) = c$.

Following the same arguments used by Debouk in [8] to illustrate the effectiveness of their protocol 3, it has been proved in [14] that under the considered architecture, the absence of failure ambiguous sequences is a necessary and sufficient condition for codiagnosability. Obviously the same result also holds under the revised definition.

Theorem 1 [14]: Consider a labeled PN system $(N, m_0, \mathcal{L})$ whose labeling function $\mathcal{L}$ is defined over an alphabet A. Assume that $(N, m_0, \mathcal{L})$ is monitored by a set $\mathcal{J} = \{1, 2, ..., \nu\}$ of local sites. The system is codiagnosable iff there do not exist failure ambiguous sequences that are arbitrarily long after the occurrence of any fault in T_f^i, for $i = 1, \ldots, r$.

IV. Extended Basis Reachability Graph

In this section we first introduce a particular graph, called *Extended Basis Reachability Graph*. Then, we prove some properties that are the starting point for the proposed approach of codiagnosability analysis. Note that Definitions 3 to 7 are taken from [15] and [6].

For the sake of simplicity, in the rest of the paper we assume that there is a single fault class T_f. In the following it is clearly discussed how to deal with the case of several fault classes.

Definition 3 [15]: Given a marking m and an observable transition t, the set of *explanations* of t at m is denoted by $\Sigma(m, t) = \{\sigma \in T_u^* \mid m[\sigma\rangle m', m'[t\rangle\}$, and the set of *e-vectors (or explanation vectors)* is denoted by $Y(m, t) = \pi(\Sigma(m, t))$.

Definition 4 [15]: Given a marking m and an observable transition t, the set of *minimal explanations* of t at m is denoted by $\Sigma_{min}(m, t) = \{\sigma \in \Sigma(m, t) \mid \nexists \sigma' \in \Sigma(m, t) : \pi(\sigma') \lneqq \pi(\sigma)\}$, and the set of *minimal e-vectors* is denoted by $Y_{min}(m, t) = \pi(\Sigma_{min}(m, t))$.

Definition 5 [15]: Let $(N, m_0, \mathcal{L})$ be a labeled PN system and $w \in L^*$ be an observation, where $N = (P, T, F, W)$ and $T = T_o \cup T_u$. We define the following sets of pairs:

$$\hat{\mathcal{J}}(w) = \{(\sigma_o, \sigma_u), \sigma_o \in T_o^*, \mathcal{L}(\sigma_o) = w, \sigma_u \in T_u^* \mid$$
$$[\exists \sigma \in \mathcal{L}^{-1}(w) : \sigma_o = P_o(\sigma), \sigma_u = P_u(\sigma)]$$
$$\wedge [\nexists \sigma' \in \mathcal{L}^{-1}(w) : \sigma_o = P_o(\sigma'), \sigma'_u = P_u(\sigma')$$
$$\wedge \pi(\sigma'_u) \lneqq \pi(\sigma_u)]\},$$

and

$$\hat{Y}_{min}(m_0, w) = \{(\sigma_o, y), \sigma_o \in T_o^*, \mathcal{L}(\sigma_o) = w, y \in \mathbb{N}^{|T_u|} \mid$$
$$\exists (\sigma_o, \sigma_u) \in \hat{\mathcal{J}}(w) : \pi(\sigma_u) = y\}.$$

In simple words, $\hat{\mathcal{J}}(w)$ is the set of pairs whose first element is the sequence $\sigma_o \in T_o^*$ labeled w and whose second element is the corresponding sequence of unobservable transitions interleaved with σ_o whose firing enables σ_o and whose firing vector is minimal. The firing vectors of these sequences are called j-vectors.

$\hat{Y}_{min}(m_0, w)$ is the set of pairs whose first element is the sequence $\sigma_o \in T_o^*$ labeled w and whose second element is the corresponding j-vector (justification vector).

Definition 6 [15]: Let $(N, m_0, \mathcal{L})$ be a labeled PN system, $w \in A^*$ be an observation and $\hat{\mathcal{J}}(w)$ be a set of pairs. The set of *basis markings* of w is denoted by

$$M_b(w) = \{m \in \mathbb{N}^{|P|} \mid m = m_0 + [N]_u \cdot \pi(\sigma_u) + [N]_o \cdot \pi(\sigma_o),$$
$$(\sigma_o, \sigma_u) \in \hat{\mathcal{J}}(w)\},$$

and the set of all basis markings is denoted by M_b, i.e., $M_b = \bigcup_{w \in A^*} M_b(w)$.

In simple words, a basis marking is a marking that can be reached from the initial marking firing a sequence of transitions that is consistent with the observation and a sequence of unobservable transitions, interleaved with the previous sequence, whose firing is strictly necessary to enable it (in the sense that its firing vector is minimal) [15]. The set of basis markings is a subset (usually a strict subset) of the set of reachable markings. Therefore, if the net is bounded, the set of basis markings is finite.

In [6] it has been proved that when performing centralized diagnosability, it is useful to compute basis markings assuming that fault transitions are observable.

Definition 7 [6]: An *extended basis marking (EBM)* is a basis marking computed assuming that all transitions in T_f are observable. The set of all EBMs is denoted by M_e.

The set M_e can be computed by restricting the minimal explanations to the set of regular unobservable transitions T_{reg}.

Example 2: Let us consider the labeled PN system in Fig. 1 previously introduced in Example 1, where $T_o = \{t_3, t_6, t_{10}\}$ and $T_f = \{t_9\}$. The set of EBMs is $\{m_i \mid m_i = [k-i\ 0\ 0\ 0\ 0\ 0\ 0\ i]^T,\ i = 0, 1, ..., k\}$.

Let us now define a graph whose nodes are uniquely associated with EBMs and edges are labeled with either observable transitions (and their labels) or with fault transitions. In the following, we denote $Y^{reg}_{min}(m, t)$ the set of minimal e-vectors restricted to T_{reg}. The set $Y^{reg}_{min}(m, t)$ can be computed using Algorithm 4.4 in [15].

Definition 8: Let $(N, m_0, \mathcal{L})$ be a labeled PN system, T_f be the set of fault transitions and M_e be the set of EBMs. The *Extended Basis Reachability Graph (EBRG)* is a (non-deterministic) finite state automaton $G_e = (M_e, E, \Delta, m_0)$, where M_e is the set of states; $E \subseteq (T_o \times A) \cup T_f$ is the set of event labels; $\Delta \subseteq M_e \times E \times M_e$ is the transition relation; and m_0 is the initial state. In particular, $(m, e, m') \in \Delta$ where $e = t(a) \in T_o \times A$ or $e = t \in T_f$, if and only if $\exists y \in Y^{reg}_{min}(m, t)$ and $m' = m + [N]_{reg} \cdot y + [N](\cdot, t)$.

Note that a similar graph, called *Modified Basis Reachability Graph* (MBRG) has been proposed in [6] to perform centralized diagnosis. In the MBRG, as well as in the EBRG, a different node is associated with each extended basis marking and edges are labeled either with an observable transition (and its label) or with a fault transition. However, the MBRG may contain a higher number of edges since a different edge is associated with each minimal e-vector. In more detail, if there exist two minimal explanations of a given transition that lead to the same extended basis marking, in the MBRG two different edges are associated with it, while only one edge appears in the EBRG.

```
Algorithm 1: [EBRG construction]
Input: A labeled PN system $(N, m_0, \mathcal{L})$.
Output: The EBRG $G_e$.
1. Let $m_0$ be the initial node.
2. While nodes with no tag exist, do
   2.1. select a node $m$ with no tag,
   2.2. for all $t \in T_o \cup T_f$, do
        • if $Y^{reg}_{min}(m, t) \neq \emptyset$, then
          – for all $y \in Y^{reg}_{min}(m, t)$, do
              let $m' = m + [N]_{reg} \cdot y + [N](\cdot, t)$,
              if $\nexists$ a node $m'$, then add a node $m'$,
              if $t \in T_o \wedge \nexists$ an arc $t(e)$ from $m$ to $m'$, where
              $e = \mathcal{L}(t)$, then add an arc $t(e)$ from $m$ to $m'$,
              if $t \in T_f \wedge \nexists$ an arc $t$ from $m$ to $m'$, then
              add an arc $t$ from $m$ to $m'$,
   2.3. tag the node $m$ "old".
3. Remove all tags.
```

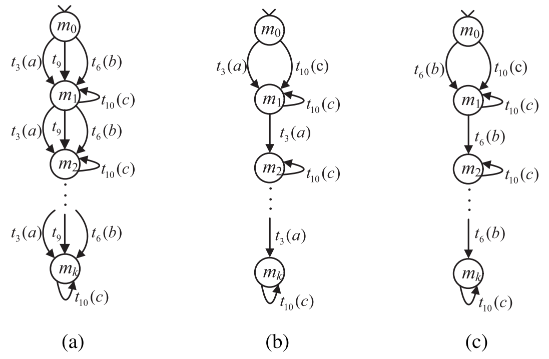

Fig. 2: a) G_e: EBRG of $(N, m_0, \mathcal{L})$, b) G^1_e: Nonfailure EBRG wrt site 1, and c) G^2_e: Nonfailure EBRG wrt site 2.

Example 3: Consider again the labeled PN system in Example 1. The EBRG G_e is shown in Fig. 2a, where $\{m_i | m_i = [k-i\ 0\ 0\ 0\ 0\ 0\ 0\ i]^T, i = 0, 1, ..., k\}$.

Property 1: Let $(N, m_0, \mathcal{L})$ be a labeled PN system, G_e be its EBRG and $L(G_e)$ be the language generated by G_e. It holds that $\rho(L(N, m_0)) = L(G_e)$, where $\rho(L(N, m_0))$ is the projection of $L(N, m_0)$ over $T_o \cup T_f$.

Proof. Let us recall Theorem 3.8 in [18] which can be restated as follows. In a net whose unobservable subnet is acyclic there exists a firing sequence $\sigma \in T^*$ such that $m_0[\sigma\rangle$ with observable projection $P_o(\sigma) = w$ and Parikh vector $\pi(\sigma)$ *if and only if* there exists a sequence σ' with $P_o(\sigma') = w$ and an unobservable sequence $\sigma_u \in T^*_u$ such that $m_0[\sigma'\rangle m[\sigma''\rangle$ where m is a basis marking and $\pi(\sigma) = \pi(\sigma') + \pi(\sigma'')$.

The result follows observing that, by construction, there exists a path in the EBRG that starts from the initial node m_0 and reaches a node m generating word w if and only if basis marking m can be reached on the net from the initial marking m_0 by a firing sequence σ' with $P_o(\sigma') = w$. □

In words, the above property claims that the set of transition sequences in $L(G_e)$ coincides with the projection of $L(N, m_0)$ over the set $T_o \cup T_f$.

V. VERIFIER

In this section, we show that the codiagnosability of a bounded PN can be checked by analyzing a special automaton called *Verifier*.

For the sake of simplicity, and without loss of generality, we assume that the PN is monitored by only two local sites.

In the following we denote by $(N', m_0, \mathcal{L}')$ the T'-induced subnet of $(N, m_0, \mathcal{L})$, where $T' = T \setminus T_f$, i.e., $(N', m_0, \mathcal{L}')$ is the *nonfailure subnet* of $(N, m_0, \mathcal{L})$. Therefore, $L(N', m_0)$ is the language composed by all sequences of $L(N, m_0)$ that do not contain faults, and $\mathcal{L}'$ is equal to $\mathcal{L}$ restricted to $T \setminus T_f$.

Definition 9: Let $(N, m_0, \mathcal{L})$ be a labeled PN system and $(N', m_0, \mathcal{L}')$ be its nonfailure subnet. The *nonfailure EBRG* wrt site j, denoted by $G_e^j = (M_j, E_j, \Delta_j, m_0)$, is the EBRG of $(N', m_0, \mathcal{L}')$ constructed under the assumption that the set of observable transitions is equal to $T_{o,j}$, and all transitions in $T' \setminus T_{o,j} = T \setminus T_f \setminus T_{o,j}$ are unobservable.

Obviously, $G_{e,j}$ can be computed using Algorithm 1 assuming that the set of observable transitions is equal to the set of transitions observable by the j-th site, namely $T_{o,j}$, and restricting minimal explanations to the set $T' \setminus T_{o,j} = T \setminus T_f \setminus T_{o,j}$.

Example 4: Consider again the Petri net in Example 1. The nonfailure-EBRGs G_e^1 and G_e^2 are shown in Fig. 2b and Fig. 2c, respectively, where $m_i = [k-i\ 0\ 0\ 0\ 0\ 0\ 0\ i]^T$, $i = 0, 1, ..., k$.

Property 2: Let $(N, m_0, \mathcal{L})$ be a labeled PN system, G_e^j be its nonfailure EBRG wrt site j and $L(G_e^j)$ be the language generated by G_e^j. It holds that $\rho_j(L(N', m_0)) = L(G_e^j)$, where $\rho_j(L(N', m_0))$ is the projection of $L(N', m_0)$ over $T_{o,j}$.

Proof. Follows the same lines of the proof of Property 1. □

We now introduce a (non-deterministic) finite state automaton, called *Verifier*, that is defined as the parallel composition of the EBRG of a given labeled PN system and the nonfailure EBRGs G_e^1 and G_e^2 of the two sites that monitor it, where synchronization is performed on the set of labels A. We denote it $V = (M^V, E^V, \Delta^V, m_0^V)$ and compute it using the following algorithm.

```
Algorithm 2: [Construction of the Verifier]
Input: $G_e = (M_e, E, \Delta, m_0)$, $G_e^1 = (M_1, E_1, \Delta_1, m_0)$ and $G_e^2 = (M_2, E_2, \Delta_2, m_0)$.
Output: The Verifier $V = (M^V, E^V, \Delta^V, m_0^V)$.
1. Let $M^V = M \times \{F,N\} \times M_1 \times M_2$.
2. Let $E^V = (T_o \cup T_f) \times (T_{o,1} \cup \{\varepsilon\}) \times (T_{o,2} \cup \{\varepsilon\})$.
3. Let $m_0^V = (m_0, N; m_0; m_0)$.
4. $\Delta^V \subseteq M^V \times E^V \times M^V$ is defined as follows:
   4.1. $((m, l; m_1, m_2)(t, \varepsilon, \varepsilon)(m', F; m_1; m_2)) \in \Delta^V$ if
        • $t \in T_f$ and $(m, t, m') \in \Delta$.
   4.2. $((m, l; m_1, m_2)(t, t_1, t_2)(m', l; m_1'; m_2')) \in \Delta^V$ if
        • $t \in T_{o,1} \cap T_{o,2}$, $(m, t, m') \in \Delta$, $(m_1, t_1, m_1') \in \Delta_1$, $(m_2, t_2, m_2') \in \Delta_2$, $\mathcal{L}_1(t) = \mathcal{L}_1(t_1)$ and $\mathcal{L}_2(t) = \mathcal{L}_2(t_2)$.
   4.3. $((m, l; m_1, m_2)(t, t_1, \varepsilon)(m', l; m_1'; m_2)) \in \Delta^V$ if
        • $t \in T_{o,1} \setminus T_{o,2}$, $(m, t, m') \in \Delta$, $(m_1, t_1, m_1') \in \Delta_1$ and $\mathcal{L}_1(t) = \mathcal{L}_1(t_1)$.
   4.4. $((m, l; m_1, m_2)(t, \varepsilon, t_2)(m', l; m_1; m_2')) \in \Delta^V$ if
        • $t \in T_{o,2} \setminus T_{o,1}$, $(m, t, m') \in \Delta$, $(m_2, t_2, m_2') \in \Delta_2$ and $\mathcal{L}_2(t) = \mathcal{L}_2(t_2)$.
5. Trim the automaton $V = (M^V, E^V, \Delta^V, m_0^V)$ by removing the states that are not reachable from the initial state $m_0^V$ and all their input and output edges.
```

Definition 10: Consider the Verifier of a given PN system. We denote as

$$\varphi : (E^V)^* \to (T_o \cup T_f)^*,$$

$$\varphi_1 : (E^V)^* \to T_{o,1}^*, \ \varphi_2 : (E^V)^* \to T_{o,2}^*$$

the three functions that assign to a generic production

$$\sigma^V = (\lambda^1, \lambda_1^1, \lambda_2^1) \dots (\lambda^k, \lambda_1^k, \lambda_2^k)$$

in the Verifier, the following three sequences of transitions:

$$\varphi(\sigma^V) = \lambda^1 \dots \lambda^k,$$

$$\varphi_1(\sigma^V) = \rho_1(\lambda_1^1 \dots \lambda_1^k), \ \varphi_2(\sigma^V) = \rho_2(\lambda_2^1 \dots \lambda_2^k),$$

where $\rho_1(\lambda_1^1 \dots \lambda_1^k)$ ($\rho_2(\lambda_2^1 \dots \lambda_2^k)$) denotes the projection of $\lambda_1^1 \dots \lambda_1^k$ ($\lambda_2^1 \dots \lambda_2^k$) over $T_{o,1}$ ($T_{o,2}$).

In simple words, given a production σ^V of the Verifier, the concatenation of the first entries of σ^V is equal to a sequence of transitions $\sigma = \lambda^1 \dots \lambda^k \in (T_o \cup T_f)^*$. Furthermore, if we denote as $\sigma_1 = \lambda_1^1 \dots \lambda_1^k$ ($\sigma_2 = \lambda_2^1 \dots \lambda_2^k$) the sequence of symbols obtained as the concatenation of the second (third) entries of σ^V, then $\varphi_1(\sigma^V)$ ($\varphi_2(\sigma^V)$) is equal to the projection of σ_1 (σ_2) over $T_{o,1}$ ($T_{o,2}$).

Given a Verifier V, we write $m^V \xrightarrow{e^V} m_1^V$ to denote that state $m_1^V \in M^V$ is reached in V from $m^V \in M^V$ with an event $e^V \in E^V$.

Theorem 2: Let $(N, m_0, \mathcal{L})$ be a labeled PN system with EBRG G_e. Let G_e^1 and G_e^2 be the nonfailure ERBGs wrt site 1 and site 2, respectively. Let V be the Verifier constructed using Algorithm 2 and $L(V) \in (E^V)^*$ its language. It holds that:

$$\sigma^V \in L(V) \Leftrightarrow$$

$$\varphi(\sigma^V) \in L(G_e), \ \varphi_1(\sigma^V) \in L(G_e^1), \ \varphi_2(\sigma^V) \in L(G_e^2), \ \text{ and}$$

$$\mathcal{L}_1(\varphi(\sigma^V)) = \mathcal{L}_1(\varphi_1(\sigma^V)), \mathcal{L}_2(\varphi(\sigma^V)) = \mathcal{L}_2(\varphi_2(\sigma^V)).$$

Proof. We prove the statement by induction on the length of a production in V.

(Basis step) The result clearly holds if we consider productions of length 0.

(Inductive step) Assume the result holds for a production $\sigma^V = e_1^V \dots e_k^V$ of length k such that

$$m_0^V \xrightarrow{e_1^V} m_1^V \dots \xrightarrow{e_k^V} m_k^V.$$

Assume that the production is continued with event $e_{k+1}^V \in E^V$, namely it is

$$m_k^V \xrightarrow{e_{k+1}^V} m_{k+1}^V.$$

Let $e_{k+1}^V = (t, \lambda_1^{k+1}, \lambda_2^{k+1})$. Four different cases may occur.

• $t \in T_f$: In such a case, according to Step 4.1 of Algorithm 2, it is $\lambda_1^{k+1} = \lambda_2^{k+1} = \varepsilon$ and consequently, $\varphi(\sigma^V e_{k+1}^V) = \varphi(\sigma^V)t$, $\varphi_1(\sigma^V e_{k+1}^V) = \varphi_1(\sigma^V)$, and $\varphi_2(\sigma^V e_{k+1}^V) = \varphi_2(\sigma^V)$. Therefore, if the result holds for σ^V, then it obviously holds for $\sigma^V e_{k+1}^V$.

• $t \in T_{o,1} \cap T_{o,2}$: In such a case, according to Step 4.2 of Algorithm 2, node m_{k+1}^V is obtained simultaneously firing at the three markings in m_k^V transitions $t, t_1, t_2 \in T_{o,1} \cap T_{o,2}$, such that $\mathcal{L}_1(t) = \mathcal{L}_1(t_1)$ and $\mathcal{L}_2(t) = \mathcal{L}_2(t_2)$, respectively (obviously, as a special case, it may also be $t = t_1 = t_2$). Therefore, it is $\varphi(\sigma^V e_{k+1}^V) = \varphi(\sigma^V)t \in L(G_e)$, $\varphi_1(\sigma^V e_{k+1}^V) = \varphi_1(\sigma^V)t_1 \in L(G_e^1)$, and $\varphi_2(\sigma^V e_{k+1}^V) = \varphi_2(\sigma^V)t_2 \in L(G_e^2)$. Finally, it is

$$\mathcal{L}_1(\varphi(\sigma^V e_{k+1})) = \mathcal{L}_1(\varphi_1(\sigma^V e_{k+1}))$$

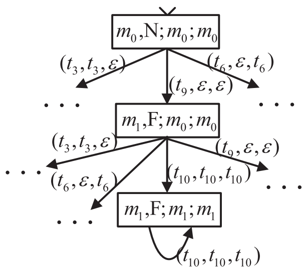

Fig. 3: The Verifier of the PN system in Example 1.

being

$$\mathcal{L}_1(\varphi(\sigma^V e_{k+1})) = \mathcal{L}_1(\varphi(\sigma^V)t) = \mathcal{L}_1(\varphi(\sigma^V))\mathcal{L}_1(t),$$

$$\mathcal{L}_1(\varphi_1(\sigma^V e_{k+1})) = \mathcal{L}_1(\varphi_1(\sigma^V)t_1) = \mathcal{L}_1(\varphi_1(\sigma^V))\mathcal{L}_1(t_1),$$

$$\mathcal{L}_1(\varphi(\sigma^V)) = \mathcal{L}_1(\varphi_1(\sigma^V))$$

by the inductive assumption. Analogously, we may prove that

$$\mathcal{L}_2(\varphi(\sigma^V e^V_{k+1})) = \mathcal{L}_2(\varphi_2(\sigma^V e^V_{k+1})).$$

• $t \in T_{o,1} \setminus T_{o,2}$: In such a case λ_1^{k+1} is equal to a transition t_1 (that may also be coincident with t) such that $\mathcal{L}_1(t) = \mathcal{L}_1(t_1)$, while $\lambda_2^{k+1} = \varepsilon$. The result follows from Step 4.3 of Algorithm 2 using arguments analogous to the previous two steps.

• $t \in T_{o,2} \setminus T_{o,1}$: In such a case $\lambda_1^{k+1} = \varepsilon$ and λ_2^{k+1} is equal to a transition t_2 (that may also be coincident with t) such that $\mathcal{L}_2(t) = \mathcal{L}_2(t_2)$. The result follows from Step 4.4 of Algorithm 2 using arguments analogous to the first two steps. □

In simple words, Theorem 2 implies that a generic production σ^V in the Verifier captures three sequences of transitions: $\sigma = \varphi(\sigma^V)$, $\sigma_1 = \varphi_1(\sigma^V)$, $\sigma_2 = \varphi_2(\sigma^V)$, which belong to $L(G_e)$, $L(G_e^1)$, and $L(G_e^2)$, respectively. Furthermore, the projections of σ and σ_1 over A_1 coincide. Analogously, the projection of σ over A_2 coincides with the projection of σ_2 over A_2. Finally, given node m^V of the Verifier reached with a certain production σ^V, label N (resp., F) in m^V indicates that the first sequence of transitions associated with σ^V (namely, $\varphi(\sigma^V)$) does not (resp., does) include a fault in T_f.

A state $(m, l; m_1; m_2)$ in the Verifier is called an l-state. For example, the initial state m_0^V is an N-state. A cycle in the Verifier is called an l-cycle if each state in the cycle is an l-state.

Example 5: Consider again the PN system in Example 1. Fig. 3 shows a part of the Verifier. The cycle $((m_1, \text{F}; m_1; m_1), (t_{10}, t_{10}, t_{10}), (m_1, \text{F}; m_1; m_1))$ is an F-cycle.

Now, since sequences in $L(G_e)$ also include faults, while sequences in $L(G_e^j)$'s do not, looking at sequences in the Verifier, we could establish if there exists any faulty sequence in $L(G_e)$ whose observable projection in all sites could be explained without the firing of any fault. This is formalized in the following result.

Property 3: Let $(N, m_0, \mathcal{L})$ be a labeled PN system with EBRG G_e. Let G_e^1 and G_e^2 be the nonfailure EBRGs wrt site 1 and site 2, respectively. The Verifier V constructed using Algorithm 2 has the following properties.

a) The language of V is:

$$\begin{aligned} L(V) = \{ \quad & \sigma^V \in (E^V)^* \mid \exists \bar\sigma, \bar\sigma_1, \bar\sigma_2 \in L(N, M_0) \\ & \varphi(\sigma^V) = \rho(\bar\sigma), \\ & \varphi_1(\sigma^V) = \rho_1(\bar\sigma_1),\ \varphi_2(\sigma^V) = \rho_2(\bar\sigma_2), \\ & \mathcal{L}_1(\bar\sigma) = \mathcal{L}_1(\bar\sigma_1),\ \mathcal{L}_2(\bar\sigma) = \mathcal{L}_2(\bar\sigma_2), \\ & \bar\sigma_1 \cap T_f = \bar\sigma_2 \cap T_f = \emptyset\} \end{aligned}$$

where $\rho(\bar\sigma)$ denotes the projection of $\bar\sigma$ over $T_o \cup T_f$.

b) If state $(m, l; m_1; m_2)$ is reached in V from the initial state with a sequence σ^V, then

$$l = \begin{cases} \text{N} & \text{iff } (\forall \bar\sigma \in \rho^{-1}(\sigma)),\ \bar\sigma \cap T_f = \emptyset \\ \text{F} & \text{iff } (\forall \bar\sigma \in \rho^{-1}(\sigma)),\ \bar\sigma \cap T_f \neq \emptyset \end{cases}$$

where $\sigma = \varphi(\sigma^V)$.

Proof. Condition a) follows from the fact that $V = G_e \parallel G_e^1 \parallel G_e^2$ is the concurrent composition of the three EBRGs on the sets, resp., $T_o \cup T_f$, $T_{o,1}$ and $T_{o,2}$. In addition, the first EBRG is obtained from net $(N, m_0, \mathcal{L})$ while the nonfailure EBRGs have been obtained from the nonfailure subnet $(N', m_0, \mathcal{L}')$: hence $\bar\sigma_1 \cap T_f = \bar\sigma_2 \cap T_f = \emptyset$.

Condition b) follows from the definition of the transition relation of V. In fact by construction $(m, \text{N}; m_1; m_2)$ is reached in V from the initial state with sequence σ^V if and only if $\varphi(\sigma^V) \cap T_f = \emptyset$ and hence by Property 1 for all $\bar\sigma \in \rho^{-1}(\sigma)$ it holds $\bar\sigma \cap T_f = \emptyset$. The same applies if we consider F-states. □

Theorem 3: Let $V = (M^V, E^V, \Delta^V, m_0^V)$ be the Verifier of a given PN system constructed by Algorithm 2. The net has failure ambiguous sequences of arbitrary length after the occurrence of some fault in T_f iff V contains F-cycles.

Proof. Consider an evolution of V such that

$$m_0^V \xrightarrow{\sigma^V} m^{V\prime} \xrightarrow{(t,\varepsilon,\varepsilon)} m^V,$$

where $m^{V\prime}$ is an N-state and m^V is an F-state. Hence there exists a sequence $\bar\sigma' \in \rho^{-1}(\varphi(\sigma^V)t)$ that ends with a fault by Property 3. By assumption A1 this sequence can be continued indefinitely. This means that all sequences in the set

$$\begin{aligned} \mathcal{A}(\bar\sigma') = \quad & \{\bar\sigma = \bar\sigma'\bar\sigma'' \mid |\bar\sigma''| \geq 0, \\ & \rho(\bar\sigma) = \varphi(\sigma^V),\ \sigma^V \in L(V)\} \end{aligned}$$

are failure ambiguous according to Definition 2. Additionally for any such sequence $\bar\sigma$, the sequence σ^V such that $\rho(\bar\sigma) = \varphi(\sigma^V)$, drives the Verifier to an F-state by Property 3.

Now there exist failure ambiguous sequences of arbitrary length after the fault, if and only if there exists a sequence $\bar\sigma'$ of the net that ends with a fault and is such that $\mathcal{A}(\bar\sigma')$ is an infinite set. From this set we can extract an infinite increasing sequence $\bar\sigma_0$, $\bar\sigma_1 = \bar\sigma_0\bar t_1$, $\bar\sigma_2 = \bar\sigma_0\bar t_1\bar t_2, \ldots$. Obviously the chain of states of the Verifier reached by sequences σ_i^V such that $\varphi(\sigma_i^V) = \rho(\bar\sigma_i)$, $i = 0, 1, \ldots$, belongs to an infinite path of F-states by assumption A3. Since the set of states of V is finite by assumption A2, this is possible if and only if there exists an F-cycle. □

Corollary 1: A labeled PN system $(N, m_0, \mathcal{L})$ monitored by two local sites is codiagnosable iff its Verifier has no F-cycles.

Proof. Straightforward from Theorems 1 and 3. □

Example 6: Let us consider again Example 5. By Corollary 1, we conclude that the PN system is not codiagnosable since there exists an F-cycle in the Verifier.

In the discussion so far, we only considered one fault class. In the case of r fault classes we need to construct r Verifiers, one for each fault class. When verifying the codiagnosability wrt T_f^i, all fault transitions in $T_f \setminus T_f^i$ should be considered as regular unobservable transitions.

We conclude this section with a brief discussion on the complexity of the proposed method. The size of the state space of the EBRG, in the worst case, is equal to that of the reachability graph. However, the EBRG has significantly fewer states than the reachability graph in most cases. For example, the number of reachable markings of the PN in Example 1 is $\binom{8+k-1}{k}$, while the number of states in the EBRGs is $k+1$.

Let x be the number of nodes in G_e, i.e., $x = |M_e|$. Assume that the PN system is monitored by ν local sites and has r fault classes. According to Algorithm 2, the number of nodes and edges in the Verifier are at most equal to $2x^{\nu+1}$ and $2x^{\nu+1} \times |T|^{\nu+1}$, respectively. Moreover, we need to check all cycles in the Verifier. This can be computed by Tarjan's strongly connected components algorithm [19], whose complexity is linear in the sum of the number of nodes and arcs in the Verifier, i.e., $O((x \times |T|)^{\nu+1})$. Hence, the overall complexity is $O((x \times |T|)^{\nu+1} \times r)$.

Therefore the advantage of the proposed approach lies in the use of basis markings rather than exhaustively enumerating the set of reachable markings. The numerical example above clearly highlights this. However, as discussed in other papers, we cannot a priori quantify such an advantage since it depends on the structure of the net, on the labeling function, and on the initial marking.

VI. K-CODIAGNOSABILITY

A labeled PN system monitored by a set of local sites is *K-codiagnosable* wrt a given fault class if faults in that class can be detected in at most K observations after the occurrence of the fault. The formal definition of K-codiagnosability follows.

Definition 11: Consider a labeled PN system $(N, m_0, \mathcal{L})$ and an integer K. Assume that $(N, m_0, \mathcal{L})$ is monitored by a set $\mathcal{J} = \{1, 2, ..., \nu\}$ of local sites. The labeled PN system $(N, m_0, \mathcal{L})$ is *K-codiagnosable* wrt the i-th fault class T_f^i if there does not exist a transition sequence σ such that: (1) $T_f^i \cap \sigma \neq \emptyset$; (2) σ is failure ambiguous wrt T_f^i; (3) the number of observable transitions in σ after the first occurrence of a fault transition $t_f \in T_f^i$ is K.
The labeled PN system $(N, m_0, \mathcal{L})$ is K-codiagnosable if it is K-codiagnosable wrt all fault classes.

Obviously, a K-codiagnosable PN system is also K'-codiagnosable if $K' > K$. In this section, we provide an algorithm to compute the smallest value of K, denoted by K_{min} such that the system is K_{min}-codiagnosable. The proposed approach is based on the notion of Verifier. Therefore, for the sake of simplicity, we present it under the assumption of a single fault class.

Based on the results in Section V, we know that a system is codiagnosable if and only if all paths in the Verifier that contain F-states end in a deadlock state. Indeed, if such is not the case, it means that there are faulty paths of infinite length and the system is not codiagnosable. Furthermore, based on the above definition, the value of K_{min} is equal to the maximum number of observable transitions after the first occurrence of a fault in a path containing F-states, plus one.

Before providing the algorithm for the computation of K_{min}, let us introduce a preliminary definition.

Definition 12: Let $V = (M^V, E^V, \Delta^V, m_0^V)$ be the Verifier of a given labeled PN system $(N, m_0, \mathcal{L})$ constructed by Algorithm 2. Let Z_F be the set of F-states of the Verifier. The *observable delay* wrt an F-state $z \in Z_F$ is defined as $od(z) = \{1 + \max |P_o(\varphi(\sigma^V))| \mid \sigma^V \in (E^V)^*, (\exists z' \in Z_F) : z \xrightarrow{\sigma^V} z'\}$.

In simple words, the observable delay of an F-state z is equal to the maximum number of transitions that could be observed starting from state z. Such a number is obviously finite if the system is codiagnosable since in such a case, a deadlock state is reached in a finite number of steps. Note that in the above definition, being $z \in Z_F$, it is also $z' \in Z_F$ since an N-state cannot be reached from an F-state.

Clearly, it is $K_{min} = \max_{z \in Z_F} od(z)$.

```
Algorithm 3: [Computation of $K_{min}$]
Input: A labeled PN system $(N, m_0, \mathcal{L})$.
Output: $K_{min}$.
1. Construct the Verifier $V$ of $(N, m_0, \mathcal{L})$ and check if
   $(N, m_0, \mathcal{L})$ is codiagnosable.
2. If $(N, m_0, \mathcal{L})$ is codiagnosable, then
   2.1. Let $k = 0$, $stop = False$, $\Gamma = \emptyset$ and $\forall z \in Z_F$ :
        $od^0(z) = 1$.
   2.2. While not $stop$, do
        • $k = k + 1$.
        • for all $z \in Z_F$, do
          – for all $z' \in Z_F$ such that $\exists(\lambda, \lambda_1, \lambda_2) \in E^V$ :
            $z \xrightarrow{(\lambda,\lambda_1,\lambda_2)} z'$, do
              if $\lambda \in T_o$, then $\gamma = od^{k-1}(z') + 1$;
              else $\gamma = od^{k-1}(z')$.
              $\Gamma = \Gamma \cup \{\gamma\}$.
          – let $od^k(z) = \max\{od^{k-1}(z), \max_{\gamma \in \Gamma} \gamma\}$.
          – let $\Gamma = \emptyset$.
        • if $\forall z \in Z_F$, $od^k(z) = od^{k-1}(z)$, then $stop =$
          $True$.
   2.3. Let $K_{min} = \max_{z \in Z_F} od^k(z)$.
```

The above algorithm could be explained as follows. The observable delay is initialized at "1" for each F-state (Step 2.1), which means that a fault is detected observing at least an

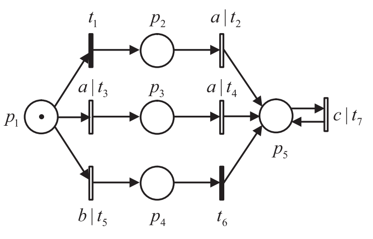

Fig. 4: A labeled PN system $(N, m_0, \mathcal{L})$.

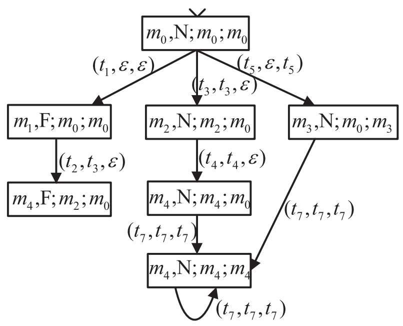

Fig. 5: The Verifier of the PN system in Example 7.

observable transition after its occurrence. Step 2.2 iteratively updates the observable delay of all the F-states counting the number of observable transitions contained in faulty paths starting from them. Step 2.2 is executed until the value of the observable delay of each F-state no longer changes. Finally, K_{min} is computed at Step 2.3 as the maximum observable delay wrt all the F-states.

Example 7: Consider the labeled PN system $(N, m_0, \mathcal{L})$ in Fig. 4, where $T_o = \{t_2 - t_5, t_7\}$, $T_u = \{t_1, t_6\}$, $T_f = \{t_1\}$ and $m_0 = [1\ 0\ 0\ 0\ 0]^T$. The labeling function is defined as follows: $\mathcal{L}(t_2) = \mathcal{L}(t_3) = \mathcal{L}(t_4) = a$, $\mathcal{L}(t_5) = b$ and $\mathcal{L}(t_7) = c$. EBMs are: $m_0 = [1\ 0\ 0\ 0\ 0]^T$, $m_1 = [0\ 1\ 0\ 0\ 0]^T$, $m_2 = [0\ 0\ 1\ 0\ 0]^T$, $m_3 = [0\ 0\ 0\ 1\ 0]^T$, $m_4 = [0\ 0\ 0\ 0\ 1]^T$. Assume that the PN is monitored by two local sites whose alphabets are equal to $A_1 = \{a, c\}$ and $A_2 = \{b, c\}$, respectively. The Verifier V of the PN system is shown in Fig. 5. By Corollary 1, the PN system is codiagnosable.

The set of F-states is $Z_F = \{(m_1, \text{F}; m_0; m_0), (m_4, \text{F}; m_2, m_0)\}$. By Algorithm 3, $od(m_1, \text{F}; m_0; m_0) = 2$ and $od(m_4, \text{F}; m_2, m_0) = 1$. Thus, $K_{min} = 2$. Hence, the labeled PN system $(N, m_0, \mathcal{L})$ is 2-codiagnosable.

In the case of r fault classes, the approach should be applied r times separately, considering one fault class at a time and the transitions in the other fault classes as regular unobservable transitions. A value of $K_{min,i}$ is computed for each fault class. The system is K_{min}-codiagnosable for $K_{min} = \max_{i=1,2,\ldots,r} K_{min,i}$.

VII. Conclusions and future work

This paper proposes a new approach to verify codiagnosability of labeled bounded Petri nets. It is based on the result that a necessary and sufficient condition for codiagnosability is the absence of failure ambiguous sequences that are arbitrarily long after the occurrence of any fault. An automaton, called Verifier, is constructed to detect the presence of such kind of sequences. The main feature of the proposed method is that it uses the notion of basis marking thus avoiding exhaustive enumeration of the state space.

Our future efforts will be twofold. First we plan to develop efficient codiagnosability analysis approaches for unbounded Petri nets. Second, we plan to study fault diagnosis in a distributed setting, namely assuming that there is no coordinator but the sites could communicate with each other according to a given communication topology network.

References

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

[2] S. Jiang, Z. Huang, V. Chandra, and R. Kumar. A polynomial algorithm for testing diagnosability of discrete-event systems. *Automatic Control, IEEE Transactions on*, 46(8):1318–1321, Aug 2001.

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

[4] M.P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu. A New Approach for Diagnosability Analysis of Petri Nets Using Verifier Nets. *Automatic Control, IEEE Transactions on*, 57(12):3104–3117, Dec 2012.

[5] G. Jiroveanu and R.K. Boel. The Diagnosability of Petri Net Models Using Minimal Explanations. *Automatic Control, IEEE Transactions on*, 55(7):1663–1668, July 2010.

[6] M.P. Cabasino, A. Giua, and C. Seatzu. Diagnosability of Discrete-Event Systems Using Labeled Petri Nets. *Automation Science and Engineering, IEEE Transactions on*, 11(1):144–153, Jan 2014.

[7] F. Basile, P. Chiacchio, and G. De Tommasi. On K-diagnosability of Petri nets via integer linear programming. *Automatica*, 48(9):2047 – 2058, 2012.

[8] R. Debouk, S. Lafortune, and D Teneketzis. Coordinated Decentralized Protocols for Failure Diagnosis of Discrete Event Systems. *Discrete Event Dynamic Systems*, 10(1):33–86, January 2000.

[9] W. Qiu and R. Kumar. Decentralized failure diagnosis of discrete event systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 36(2):384–395, March 2006.

[10] X. Yin and S. Lafortune. Codiagnosability and coobservability under dynamic observations: Transformation and verification. *Automatica*, 61:241 – 252, 2015.

[11] W. Wang, A. R. Girard, S. Lafortune, and F. Lin. On Codiagnosability and Coobservability With Dynamic Observations. *IEEE Transactions on Automatic Control*, 56(7):1551–1566, July 2011.

[12] S. Takai and T. Ushio. Verification of Codiagnosability for Discrete Event Systems Modeled by Mealy Automata With Nondeterministic Output Functions. *IEEE Transactions on Automatic Control*, 57(3):798–804, March 2012.

[13] F. Basile, P. Chiacchio, and G. De Tommasi. Decentralized K-Diagnosability of Petri Nets. *IFAC Proceedings Volumes*, 45(29):214 – 220, 2012.

[14] M.P. Cabasino, A. Giua, A. Paoli, and C. Seatzu. Decentralized diagnosability analysis of discrete event systems using Petri nets. In *Proc. 18th IFAC World Congr.*, volume 18, pages 6060–6066, Aug 2011.

[15] 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, Sep 2011.

[16] N. Ran, H. Su, A. Giua, and C. Seatzu. Codiagnosability verification of bounded Petri nets using basis markings. In *2016 IEEE 55th Conference on Decision and Control*, pages 3948–3953, Dec 2016.

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

[18] M.P. Cabasino, A. Giua, and C. Seatzu. Fault Detection for Discrete Event Systems using Petri Nets with Unobservable Transitions. *Automatica*, 46(9):1531–1539, Sep 2010.

[19] R. Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.