Diagnosability analysis of labeled Time Petri net systems

Francesco Basile, *Senior Member, IEEE,* Maria Paola Cabasino, Carla Seatzu, *Senior Member, IEEE*

***Abstract*—In this paper we focus on two notions of diagnosability for labeled Time Petri net systems: K-diagnosability implies that any fault occurrence can be detected after at most K observations, while τ-diagnosability implies that any fault occurrence can be detected after at most τ time units. A procedure to analyze such properties is provided. The proposed approach uses the Modified State Class Graph, a graph the authors recently introduced for the marking estimation of labeled Time Petri net systems, which provides an exhaustive description of the system behavior. A preliminary diagnosabilty analysis of the underlying logic system based on classical approaches taken from the literature is required. Then, the solution of some linear programming problems should be performed to take into account the timing constraints associated with transitions.**

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

I. Introduction

The explicit consideration of time is crucial for the specification and the verification of systems such as transportation systems [5], communication protocols, circuits, or real-time systems as well as to study a series of extremely important problems such as state estimation, state feedback control and fault diagnosis.

In the Petri net (PN) framework the first main distinction is between Time PNs [23] and Timed PNs [20]. In Time PNs enabled transitions may fire within given time intervals that may either be associated with places or transitions (P-Time PNs and T-Time PNs, respectively). In Timed PNs enabled transitions fire as soon as given time delays have elapsed. As in the previous case, delays may either be associated with places or transitions (P-Timed PNs and T-Timed PNs, respectively). In this paper we consider T-Time PNs and we call them Time PNs (TPNs) for short.

In [4] we solved the problem of *marking* estimation of a labeled TPN. We first presented an algorithm for the construction of a graph called *Modified State Class Graph* (MSCG) that collects all the information on the evolution of the labeled TPN system; then, given an observation and a time instant we illustrated a procedure based on the exploration of the MSCG and on the solution of some linear programming problems (LPPs), that allows one to determine the set of markings consistent with the observation and the considered time instant. Finally, we present a procedure to perform fault diagnosis using the MSCG.

In this paper we analyze K-diagnosability and τ-diagnosability of labeled TPNs. The problem of diagnosability consists in determining a priori if a system is diagnosable, i.e., if it is possible to reconstruct the occurrence of fault events observing words of finite length. When we deal with TPNs two different diagnosability problems can be solved: After how many *observations* are we able to detect a fault occurrence? Or after how many *time units* are we able to detect a fault occurrence? In the first case we study K-diagnosability, while in the second case we study τ-diagnosability of the system. Specifically, a TPN is said K*-diagnosable* if once a fault has occurred its occurrence can be detected after at most K observations. On the other hand, a TPN is said τ*-diagnosable* if once a fault has occurred its occurrence can be detected after at most τ time units.

In this paper, in accordance with most of the literature in this area, we assume that only a subset of transitions can be observed, e.g., because a sensor that produces an observable output when they fire is associated with them. We call such transitions *observable*. The other transitions are called *unobservable* (or *silent*) because their firing cannot be observed. We consider fault transitions as a subset of unobservable transitions. Moreover, we assume that the same sensor can be associated with more than one transition, i.e., more than one transition may share the same label. We call such transitions *indistinguishable*.

The main contribution of this paper consists in a procedure to analyze K-diagnosability and τ-diagnosability. The basic idea behind the proposed approach can be summarized as follows.

The MSCG is first computed. Then, based on such a graph, a preliminary investigation based on classical approaches from the literature is carried out to establish if the underlying untimed PN system is diagnosable. If it is diagnosable, for sure the timed model is diagnosable as well. If it is not diagnosable, a further investigation should be carried out because it could happen that the information originating from time make the system diagnosable. In more detail, an approach based on the

F. Basile is with the Department of Computer Engineering, Electrical Engineering and Applied Mathematics, University of Salerno, Italy (fbasile@unisa.it)

M.P. Cabasino is with the Department of Electrical and Electronic Engineering, University of Cagliari, Italy (e-mail: cabasino@diee.unica.it).

C. Seatzu is with the Department of Electrical and Electronic Engineering, University of Cagliari, Italy (e-mail: seatzu@diee.unica.it)

M.P. Cabasino, gratefully acknowledges Sardinia Regional Government for the financial support of her Post Doc fellowship (P.O.R. Sardegna F.S.E. Operational Programme of the Autonomous Region of Sardinia, European Social Fund 2007-2013 - Axis IV Human Resources, Objective l.3, Line of Activity l.3.1.). This work also falls under the Cyprus Research Promotion Foundation (CRPF) Framework Programme for Research, Technological Development and Innovation 2009–2010 (CRPF's FP 2009–2010), co-funded by the Republic of Cyprus and the European Regional Development Fund, and specifically under Grant $T\Pi E/OPIZO/0609(BE)/08$. This work has also been partially supported by RAS project (L.R. n. 7/2007, Year 2010). Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of CRPF or RAS.

solution of an LPP is proposed. In very simple words, the LPP enables us to establish if two sequences that produce the same observations in the same time instants, one containing a fault and the other one not, may actually fire, in the sense that the unobservable events they contain are consistent with the timing constraints associated with transitions that generate them.

Note that a preliminary and partial version of this paper has been presented in [3], where only K-diagnosability was considered.

We believe that this paper provides an important contribution to the discrete event systems (DES) theory, and particularly to the PN literature, for two main reasons.

First, while the problems of diagnosis and diagnosability have been extensively studied using logic DES [24], [8], [6], [13], [17], there are relatively few works dealing with these topics in the *timed* framework, both in the case of automata [11], [18], [26], [16] and PNs [19], [28], [10], [22].

In particular, to the best of our knowledge, this is the first time that problems of K-diagnosability and τ-diagnosability using labeled TPNs are addressed in the literature.

Second, this paper is in line with our idea of building, as well as we did for untimed Petri nets [1], [12], [14], [13], [15], a research line that goes from state-estimation/fault diagnosis of labeled Time Petri nets [4] to diagnosability analysis, up to methods that make diagnosable a non-diagnosable time system and/or optimally select sensors for ensuring diagnosability (see [13] for untimed systems).

We conclude this section with a brief overview of those few papers dealing with a problem statement closely related to the one at hand.

Hashtrudi Zad *et al.* [18] introduced the concept of time-diagnosability in the context of timed DES modeled by automata, containing in their events set an event which is the tick of a global clock. The tick is assumed to be observable. Time-diagnosability requires that the occurrence of the generic fault f_i can be detected and isolated after the occurrence of at most τ_i ticks. Hence, in the context of timed DES the concept of time-diagnosability is close to K-diagnosability introduced in this paper. However there are two main differences: First, the timed DES model in [18] deals with activities with a fixed time delay, while here time intervals are considered; second, our approach relies on linear programming that is a standard and efficient tool, while [18] considers an exhaustive search on the graph. Finally, the use of linear programming also opens the door to a future research on how to make a system K-diagnosable/τ-diagnosable (if possible) acting on the transitions timing.

Another important contribution in the framework of timed automata has been proposed by Tripakis in [26] where timed automata with guards [2] are considered and the notion of δ-diagnosability (basically coincident with our definition of τ diagnosability) is proposed and shown to be PSPACE-complete in the case of $\delta \in \mathbb{Q}$. The approach proposed by Tripakis to analyze δ-diagnosability is based on the region graph tool introduced in [2]. A main difference between such an approach and the approach in this paper is that the analysis based on the region graph tool simultaneously checks logical ambiguities (two sequences have the same observable projection, one containing the fault and the other one not) and timing ambiguities (observable events may occur in the same time instants). On the contrary, our approach first focuses on logical ambiguities, and only in the case of a positive answer to this issue, check for timing ambiguities. Finally, another advantage of the approach presented in this paper is that it provides a unified framework to study both K- and τ-diagnosability.

By the way it is important to note that the approach in [26] cannot be applied to the MSCG even if the MSCG is an automaton with guards. Indeed guards in the MSCG are in general not defined by constant numbers (as in [26]) but are functions of parameters defining guards in other edges. An example that clearly points out this is provided in [4] where the MSCG has been introduced first.

A. Structure of the Paper

Section II provides some background on labeled Time Petri nets. Section III illustrates the two problem statements considered in this paper and the assumptions on which they are based. Some results that are useful in the following sections are also proved here. Preliminary results on the Modified State Class Graph are presented in Section IV. Sections V and VI explain how to analyze K-diagnosability and τ-diagnosability, respectively. Finally, conclusions are drawn in Section VIII where future lines of research in this framework are also illustrated.

II. Time Petri nets with labels

A. Background on Time Petri nets

A TPN is a Petri net also called *Place/Transition net* (P/T net) where timing is associated with transition firing [21]. In particular, time intervals are associated with transitions: A transition may fire at a given time instant if and only if it has remained logically enabled for an amount of time within its own time interval. More details are provided in the following.

As in P/T nets the net structure is defined as a quadruple $N = (P, T, Pre, Post)$ 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., the net state) is a vector $M : P \to \mathbb{N}$ that assigns to each place a nonnegative integer number of tokens, represented by black dots. We denote $M(p)$ the marking of place p.

A transition t is said to be *logically* enabled at M iff $M \geq Pre(\cdot, t)$. The firing of a transition t at a marking M yields to marking $M' = M + C(\cdot, t)$. We denote $\mathcal{A}(M)$ the set of transitions logically enabled at M, i.e., $\mathcal{A}(M) = \{t \in T \mid M \geq Pre(\cdot, t)\}$.

A *Time PN* is defined as a couple $N_d = (N, Q)$ where $N = (P, T, Pre, Post)$ defines the net structure and $Q : T \to \mathbb{Q} \times (\mathbb{Q} \cup \infty)$ defines the set of *static* intervals associated with transitions. In particular, given a transition $t_i \in T$, the function

Q associates two rational numbers with t_i (the second one may also be ∞), namely $Q(t_i) = (l_i, u_i)$, where

$$l_i \geq 0, \quad u_i \geq l_i, \quad l_i \neq \infty.$$

Transition t_i may fire iff it remains logically enabled for a time interval included in $[l_i, u_i]$. The logical enabling condition must hold consecutively (enabling memory policy) or may not hold consecutively (total memory policy), depending on the considered enabling policy [25].

A TPN N_d with a marking M_0 at the initial time instant $\tau_0 = 0$ is called a *marked* TPN, or a TPN *system*, and is denoted $\langle N_d, M_0\rangle$.

A TPN evolution is defined by a *time-transition sequence* (TTS), namely a sequence of pairs (transition, time instant), that specifies the sequence of transitions that have fired, and the corresponding time instants. As an example, the firing of $\sigma = (t_{i_1}, \tau_1)(t_{i_2}, \tau_2)\ldots(t_{i_k}, \tau_k) \in (T \times \mathbb{R}_0^+)^*$ at the initial marking M_0 means that transition with index i_1 has fired at time τ_1, transition with index i_2 has fired at time τ_2, and so on. We denote this in a compact form as $M_0[t_{i_1}(\tau_1)\rangle M_1[t_{i_2}(\tau_2)\rangle M_2 \cdots [t_{i_k}(\tau_k)\rangle M_k$, where obviously it holds $\tau_1 \leq \tau_2 \leq \cdots \tau_k$. More concisely, we also write $M_0[\sigma\rangle M_k$ or simply $M_0[\sigma\rangle$ if we do not want to specify the marking reached after the firing of σ, but only want to denote that σ may fire at M_0.

Given a TTS $\sigma = (t_{i_1}, \tau_1)(t_{i_2}, \tau_2)\ldots(t_{i_k}, \tau_k) \in (T \times \mathbb{R}_0^+)^*$, we denote as $log(\sigma) = t_{i_1} t_{i_2} \ldots t_{i_k}$ the logic sequence of transitions associated with σ, neglecting the time instants at which they have fired.

A marking M is *reachable* in $\langle N_d, M_0\rangle$ if there exists a TTS σ such that $M_0\ [\sigma\rangle\ M$. The set of all markings reachable from M_0 defines the *time reachability set* of $\langle N_d, M_0\rangle$ and is denoted by $R_t(N_d, M_0)$. Note that $R_t(N_d, M_0)$ is always a subset (usually a strict subset) of the reachability set of the underlying untimed PN [21].

A TPN system $\langle N_d, M_0\rangle$ is *bounded* if there exists a positive constant k such that, for all $M \in R_t(N_d, M_0)$, $M(p) \leq k$. By virtue of the above consideration, it obviously may happen that a TPN system is bounded even if the underlying untimed PN is unbounded.

Finally, the *enabling degree* of a transition t logically enabled at a marking M is the highest integer number k such that $M \geq k\ Pre(\cdot, t)$. In simple words, in a purely logic model, the enabling degree of a transition t at a marking M denotes how many times t may fire at M.

In the rest of the paper we assume that the considered TPNs follow a *single server semantics* and an *enabling memory policy*. More details on this can be found in [25], [9]. In simple words, when using a single server semantic each transition represents an operation that can be executed by a single operation unit (a single server). Therefore, regardless of the current enabling degree, a transition may only fire once at a time. The enabling memory policy implies that a transition has no memory of any previous enabling, i.e., if it remains enabled for a certain time and some other transition fires disabling it, when it is enabled again it does not keep into account the time intervals in which it has already been enabled. Assume as an example that a transition models the execution of a given operation that requires a time interval defined by its lower and upper bound. If such an operation is interrupted, it should start from the beginning, i.e., the time required to perform the given operation should be consecutive.

B. *Labeling function and time-label sequences*

A *labeling function* $\mathcal{L} : T \rightarrow L \cup \{\varepsilon\}$ assigns to each transition $t \in T$ either a symbol from a given alphabet L or the empty string ε.

We denote as 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 as 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 $\gamma \in L$ 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., $\mathcal{L}(t_1) = \mathcal{L}(t_2) = \gamma \in L$.

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

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

Furthermore, to avoid introducing too many different notations, we also extend the labeling function to TTSs $\sigma \in (T \times \mathbb{R}_0^+)^*$. In particular, we denote as $\mathcal{L}(\sigma) \in (L \times (\mathbb{R}_0^+ \cup \emptyset))^*$ the observable projection of σ that only contains those pairs (label-time instant) relative to observable transitions. If no pair contains observable transitions, one has $\mathcal{L}(\sigma) = (\varepsilon, \emptyset)$, i.e., the observable projection coincides with the pair (empty word-empty set) since we have no information on the transitions that have fired and on their firing time instants.

III. Problem statement and preliminary results

In this section we formalize the problem statement and clarify the assumptions on which our approach is based. Some results that will be useful in the following are also proved.

We assume that 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 (namely transitions modeling a normal behavior). The set T_f is further partitioned into r different subsets T_f^i, where $i = 1, \ldots, r$, that model different fault classes.

We make the following assumptions whose importance will be clarified in the rest of the paper.

(A1) The considered TPN is *bounded*.
(A2) For all $t \in T$, the bounds l and u associated with t are rational numbers.

A. *K-diagnosability*

Given a TPN system and an integer K, we say that the system is K-diagnosable with respect to (w.r.t.) a given fault class if it is possible to detect the occurrence of some fault in that class in at most K observations after the occurrence of the fault.

The above definition can be formalized as follows.

***Definition** 1:* Given a labeled TPN and an integer K, a net system is K-diagnosable with respect to fault class T_f^i if there do not exist two time-transition sequences σ' and σ'' such that:

- σ' and σ'' have the same observable projection, i.e., $\mathcal{L}(\sigma') = \mathcal{L}(\sigma'')$;
- there exists a fault transition $t_f \in T_f^i$ such that $t_f \in log(\sigma'')$ and $\forall t \in T_f^i$, $t \notin log(\sigma')$;
- in σ'' the number of observable events after the first occurrence of t_f is K.

A TPN system is K-diagnosable if it is K-diagnosable with respect to all fault classes. ■

An analogous definition of K-diagnosability for logic PN systems can be found in [30].

Remark. Other papers [24], [13], [7] in the framework of purely logic DES provide a slightly different definition of K-diagnosability. In particular, according to [24], [13], [7] a system is K-diagnosable w.r.t. a given fault class if it is possible to detect the occurrence of some fault in that class in at most K events occurrences (observable and unobservable) after the fault. In this paper we prefer to focus on the definition in [30] since we believe that such a definition better describes real practical problems. The following simple example clarifies this. For simplicity of explanation, let us consider the case of a timed LPN with a single fault f. Assume that a sequence $\bar{\sigma}(a, \tau_a)$ is observed and we know that no fault has occurred while observing $\bar{\sigma}$. Furthermore, assume that the only sequence that may actually occur after the observation of $\bar{\sigma}$ is $(f, \tau_f)(\varepsilon_1, \tau_1)(a, \tau_a)$, where obviously $\tau_f \leq \tau_1 \leq \tau_a$. Finally, assume that the system is 1-diagnosable according to the definition that counts both observable and unobservable events, then if the diagnoser computed the diagnosis state at time τ_1, it would detect the occurrence of the fault. However, since in practice the diagnoser does not compute the diagnosis state with continuity, and it does not know when unobservable events occur, the fault is detected at time τ_a when the diagnosis state is surely computed. The same obviously happens if the system is 1-diagnosable according to Definition 1. Based on the above consideration, we believe that our definition better characterizes what can be effectively implemented in practice, in the sense that a more evident physical meaning can be associated with K. Indeed, based on such a definition $K-1$ is the maximum number of events that could be observed after a fault, without detecting it; when the K-th observable event occurs, for sure the fault is detected.

We finally observe that Definition 1 is the counterpart of the definition of τ-diagnosability formally introduced in the following subsection. Indeed, if a system is τ-diagnosable, when τ time units have elapsed after the occurrence of a fault, for sure the fault is detected. Therefore, in both cases the diagnoser knows exactly how to interpret the diagnosis state. In one case because he measures the time, in the other case because he counts the events he has observed. ■

The following proposition relates the K-diagnosability of a TPN system with the K-diagnosability of the underlying logic PN system [30].

***Proposition** 2:* Let $\langle N_d, M_0\rangle$, with $N_d = (N, Q)$, be a labeled TPN system with labeling function $\mathcal{L}$. Let $\langle N, M_0\rangle$ be the underlying logic labeled PN system. If $\langle N, M_0\rangle$ is K-diagnosable, then there exists a $K' \leq K$ such that $\langle N_d, M_0\rangle$ is K'-diagnosable.

Proof: It follows from the fact that the effect of the timing structure can only be that of disabling sequences that are enabled in the underlying logic model. On the contrary, it may not occur that sequences that are firable in the TPN are not firable in the underlying logic model.

□

Obviously if a Petri net system (either a logic PN or TPN) is K-diagnosable, then it is also K'-diagnosable for any $K' \geq K$.

In this paper we provide a procedure to analyze K-diagnosability of TPN systems under the following additional assumption.

(A3) There do not exist cycles of unobservable transitions.

B. τ-diagnosability

Given a TPN system and a positive real number $\tau \in \mathbb{R}^+$, we say that the system is τ-diagnosable w.r.t. a given fault class if it is possible to detect the occurrence of some fault in that class in at most τ time units after the occurrence of the fault.

The above definition can be formalized as follows.

***Definition** 3:* Given a labeled TPN and a positive real number $\tau \in \mathbb{R}^+$, a net system is τ-diagnosable with respect to fault class T_f^i if there do not exist two time-transition sequences σ' and σ'' such that:

- σ' and σ'' have the same observable projection, i.e., $\mathcal{L}(\sigma') = \mathcal{L}(\sigma'')$;
- there exists a fault transition $t_f \in T_f^i$ such that $t_f \notin log(\sigma')$, and $t_f \in log(\sigma'')$;
- the duration in time of σ'' after the first occurrence of a fault transition in T_f^i is τ.

A TPN system is τ-diagnosable if it is τ-diagnosable with respect to all fault classes. ■

The following proposition relates the τ-diagnosability of a TPN system with the diagnosability of the underlying logic PN system.

***Proposition** 4:* Let $\langle N_d, M_0\rangle$, with $N_d = (N, Q)$, be a labeled TPN system with labeling function $\mathcal{L}$. Let $\langle N, M_0\rangle$ be the underlying logic labeled PN system. If $\langle N, M_0\rangle$ is diagnosable, then $\exists \bar{\tau} \in \mathbb{R}^+$ such that $\langle N_d, M_0\rangle$ is τ-diagnosable for any $\tau \geq \bar{\tau}$.

Proof: It follows the same reasoning used in the proof of Proposition 2. In fact, the effect of the timing structure can only be that of disabling sequences that are enabled in the underlying logic model. On the contrary, it may not occur that sequences that are firable in the TPN are not firable in the underlying logic PN. Thus if the underlying logic PN system is diagnosable, there always exists a positive real number $\bar{\tau}$ such that the TPN system is $\bar{\tau}$-diagnosable, thus it is also τ-diagnosable for any $\tau \geq \bar{\tau}$.

□

In this paper we provide a procedure to analyze τ-diagnosability under the following additional assumption.

(A3') There do not exist cycles of transitions (observable and unobservable) the sum of whose lower bounds is equal to zero.

As it will be better explained in Section VI, this assumption guarantees a stop condition to the proposed approach.

IV. Modified State Class Graph

The approach here presented adopts a graph called Modified State Class Graph (MSCG) that we recently presented in [4]. The main feature of the MSCG is that it collects all the information on the evolution of the TPN system. In this section we provide its definition and the algorithm for its construction.

Let us consider a labeled TPN system $\langle N_d, M_0 \rangle$, where $N_d = (N, Q)$, $N = (P, T, Pre, Post)$, $Q : T \to \mathbb{Q} \times \mathbb{Q}$. Let $\mathcal{L} : T \to L \cup \{\varepsilon\}$ be its labeling function defined over an alphabet L. The *Modified State Class Graph* is a directed graph whose nodes are called *classes*. With each class is associated a *set of states* of the net, namely a reachable marking $M \in R_t(N_d, M_0)$ and a set of inequalities Θ that define the timing constraints relative to all transitions enabled at M. Edges are labeled as $(t, \gamma, \Delta \in [l^*, u^*])$, where $t \in T$ is the transition whose firing leads from the marking in the tail node to the marking in the head node; $\gamma = \mathcal{L}(t)$ is the label associated with t, $\Delta \in [l^*, u^*]$ is a constraint on the time to go from the tail node to the head node of the edge, $l^* \leq l$ and $u^* \leq u$.

The construction of the MSCG first requires the definition of the *Modified State Class Tree* (MSCT) according to Algorithm 1. The MSCG can be immediately obtained from the MSCT simply merging duplicate nodes corresponding to equivalent classes.

In [4] we proved that under Assumptions A1 and A2 the number of classes of the MSCG is finite.

Example 5: Let us consider the labeled TPN system in Fig. 1. It is: $T_o = \{t_1, t_2, t_3\}$, $T_u = \{t_4, t_5, t_6, t_7\}$, $\mathcal{L}(t_1) = a$, and $\mathcal{L}(t_2) = \mathcal{L}(t_3) = b$. There are two fault classes: $T_f^1 = \{t_5\}$ and $T_f^2 = \{t_7\}$. Moreover, it holds: $Q(t_1) = Q(t_2) = Q(t_3) = Q(t_5) = (1, 2)$, $Q(t_4) = (2, 2)$, $Q(t_6) = (0, 2)$, and $Q(t_7) = (1, 1)$. Finally, the initial marking is $M_0 = [1\ 0\ 0\ 0\ 0]^T$.

The corresponding MSCG is shown in Fig. 2. Each class is labeled with a reachable marking $M \in R_t(N_d, M_0)$ and a set of inequalities Θ that characterize the timing intervals of the transitions enabled at that class. As an example, the initial marking M_0 and the set of timing constraints $\Theta_0 = \{2 \leq \theta_4 \leq 2,\ 1 \leq \theta_5 \leq 2\}$ are associated with the initial class C_0. In fact, M_0 enables t_4 and t_5, and it is $Q(t_4) = (2, 2)$, $Q(t_5) = (1, 2)$.

Class C_1 is labeled with marking $M_1 = [0\ 1\ 0\ 0\ 0]^T$ resulting from the firing of t_4 at M_0. At marking M_1 only t_1 is enabled. Therefore this class is characterized by the timing constraints $\Theta_1 = \{1 \leq \theta_1 \leq 2\}$.

Moreover, according to Algorithm 1, each edge of the MSCG is labeled as "$t_i, \mathcal{L}(t_i), \Delta_i \in [max\{0, l_i^k\}, min_{j:\ t_j \in \mathcal{A}(M_k)}\{u_j^k\}]$". As an example, the edge directed from C_2 to C_3 is labeled "$t_6, \varepsilon, \Delta_6 \in [0, 2]$", since the minimum of the upper bounds of the set of transitions logically enabled at C_2 is 2. ■

Algorithm 1: Construction of the Modified State Class Tree

```
input : A labeled TPN system.
output: Its Modified State Class Tree.
Initialize: The root node $C_0$ is labeled with the initial
  marking $M_0$ and a set of inequalities $\Theta_0$ defined as
  follows: $\forall t_i \in \mathcal{A}(M_0)$, let $l_i^0 \leq \theta_i \leq u_i^0$ where $l_i^0 = l_i$
  and $u_i^0 = u_i$. Tag the root node "new".
while a node tagged "new" exists do
    Select a node $C_k$ tagged "new".
    for all $t_i \in \mathcal{A}(M_k)$ do
        if $max\{0, l_i^k\} \leq min_{j:\ t_j \in \mathcal{A}(M_k)}\{u_j^k\}$, where $l_i^k$
        ($u_j^k$) is the lower (upper) bound associated with
        $t_i$ ($t_j$) at class $C_k$, then
            Let $M_q = M_k + C(\cdot, t_i)$ be the marking
            reached from $M_k$ firing $t_i$.
            for all transitions $t_r \in \mathcal{A}(M_q)$ do
                if $t_r \in \mathcal{A}(M_k)$, i.e., if $t_r$ was already
                enabled at class $C_k$, and
                $M_k - Pre(:, t_i) \geq Pre(:, t_r)$, then
                    let
                        $l_r^q = l_r^k - \Delta_i, \quad u_r^q = u_r^k - \Delta_i$
                else
                    let
                        $l_r^q = l_r, \quad u_r^q = u_r.$
            Add a new node $C_q$ labeled with marking $M_q$
            and a set of inequalities $\Theta_q$ defined as
            follows: $\forall t_r \in \mathcal{A}(M_q)$, let
            $max\{0, l_r^q\} \leq \theta_r \leq u_r^q$.
            Add an edge from $C_k$ to $C_q$ labeled
            "$t_i, \mathcal{L}(t_i),\ \Delta_i \in$
            $[max\{0, l_i^k\}, min_{j:\ t_j \in \mathcal{A}(M_k)}\{u_j^k\}]$".
            if there already exists a node equivalent to
            $C_q$ in the tree then
                tag node $C_q$ "duplicate",
            else
                tag it "new".
    Untag node $C_k$.
```

V. K-diagnosability Analysis

The procedure we propose to analyze the K-diagnosability of a TPN system can be summarized as follows.

Algorithm 6: [Analysis of K-diagnosability]

Inputs: A labeled TPN system $\langle N_d, M_0 \rangle$ with labeling function $\mathcal{L}$, fault classes T_f^i, $i \in \{1, \ldots, r\}$, and an integer number K.

Output: $DiagK_i \in \{0, 1\}$, $i \in \{1, \ldots, r\}$: $DiagK_i = 1$ ($DiagK_i = 0$) means that the system is (is not) K-diagnosable w.r.t. fault class T_f^i.

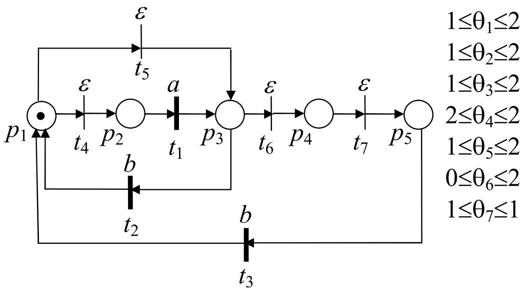

Fig. 1. The TPN considered in Example 5.

1) Build the MSCG according to Algorithm 1 and let $DiagK_i = 1$, for all $i \in \{1, \dots, r\}$.
2) Compute the nondeterministic automaton G isomorphic to the MSCG. Each state of G corresponds to a class of the MSCG. Edges are labeled with the second element of the edges of the MSCG, namely the label associated with transitions. In the case of silent transitions modeling a fault, it is also specified to which fault class transitions belong.
3) For any fault class $i \in \{1, \dots, r\}$ compute all pairs of sequences in G having the same observable projection, one containing a fault transition in T_f^i and the other one not, and such that in the faulty sequence, the number of observable events after the fault is K.
4) *If* there are no such sequences for any fault class $i \in \{1, \dots, r\}$, *then* the system is K-diagnosable.
Else if for some fault class $i \in \{1, \dots, r\}$ Step 3 provides some pair, *then* consider all such pairs and go to Step 5.
5) Given a pair of sequences in G and a fault class T_f^i, find the corresponding timed sequences $\sigma' = (t_{i'_1}, \tau'_1) \dots (t_{i'_{k'}}, \tau'_{k'})$ and $\sigma'' = (t_{i''_1}, \tau''_1) \dots (t_{i''_{k''}}, \tau''_{k''})$ in the MSCG. Let

$$\pi' =$$

$$C_{q_0} \xrightarrow{\substack{t_{i'_1}, \mathcal{L}(t_{i'_1}), \\ \Delta'_1 \in [max\{0, l^{q_0}_{i'_1}\}, min_{j:\ t_j \in \mathcal{A}(M_{q_0})}\{u^{q_0}_j\}]}}$$

$$\dots \qquad \dots$$

$$C_{q_{k'-1}} \xrightarrow{\substack{t_{i'_{k'}}, \mathcal{L}(t_{i'_{k'-1}}), \\ \Delta'_{k'} \in [max\{0, l^{q_{k'-1}}_{i'_{k'}}\}, \\ min_{j:\ t_j \in \mathcal{A}(M_{q_{k'-1}})}\{u^{q_{k'-1}}_j\}]}}$$

$$C_{q_{k'}}$$

be the path associated with sequence σ' in the MSCG.

Analogously, let

$$\pi'' =$$

$$C_{q_0} \xrightarrow{\substack{t_{i''_1}, \mathcal{L}(t_{i''_1}), \\ \Delta''_1 \in [max\{0, l^{q_0}_{i''_1}\}, min_{j:\ t_j \in \mathcal{A}(M_{q_0})}\{u^{q_0}_j\}]}}$$

$$\dots \qquad \dots$$

$$C_{q_{k''-1}} \xrightarrow{\substack{t_{i''_{k''}}, \mathcal{L}(t_{i''_{k''-1}}), \\ \Delta''_{k''} \in [max\{0, l^{q_{k''-1}}_{i''_{k''}}\}, \\ min_{j:\ t_j \in \mathcal{A}(M_{q_{k''-1}})}\{u^{q_{k''-1}}_j\}]}}$$

$$C_{q_{k''}}$$

be the path associated with sequence σ''.
Given a class C_q we denote $out(C_q)$ the set of transitions associated with edges that exit from C_q. We associate with the two paths π' and π'' the following set of constraints:

$$\left\{\begin{array}{lr}
\sum_{l=1}^{k'} \Delta'_l \leq \bar{\tau}, & (a1) \\
\bar{\tau} - \sum_{l=1}^{k'} \Delta'_l < \min_{r:t_r \in out(C_{q_{k'}})} \{u^{q_{k'}}_r\}, & (a2) \\
\sum_{l=1}^{k''} \Delta''_l \leq \bar{\tau}, & (a3) \\
\bar{\tau} - \sum_{l=1}^{k''} \Delta''_l < \min_{r:t_r \in out(C_{q_{k''}})} \{u^{q_{k''}}_r\}, & (a4) \\
\Delta'_l \geq \max\{0, l^{q_{l-1}}_{i_l}\}, & l = 1, \dots, k', \\
 & (b1) \\
\Delta'_l \leq \min_{j:t_j \in \mathcal{A}(M_{q_{l-1}})} \{u^{q_{l-1}}_j\}, & l = 1, \dots, k', \\
 & (b2) \\
\Delta''_l \geq \max\{0, l^{q_{l-1}}_{i_l}\}, & l = 1, \dots, k'', \\
 & (b3) \\
\Delta''_l \leq \min_{j:t_j \in \mathcal{A}(M_{q_{l-1}})} \{u^{q_{l-1}}_j\}, & l = 1, \dots, k'', \\
 & (b4) \\
\sum_{l=1}^{q'} \Delta'_l = \sum_{l=1}^{q''} \Delta''_l, & \forall t_{i'_{q'}}, t_{i''_{q''}} \in T_o \text{ with} \\
 & \text{same label and same} \\
 & \text{observable prefix} \\
 & (c) \\
\bar{\tau} \geq 0 & (d)
\end{array}\right. \quad (1)$$

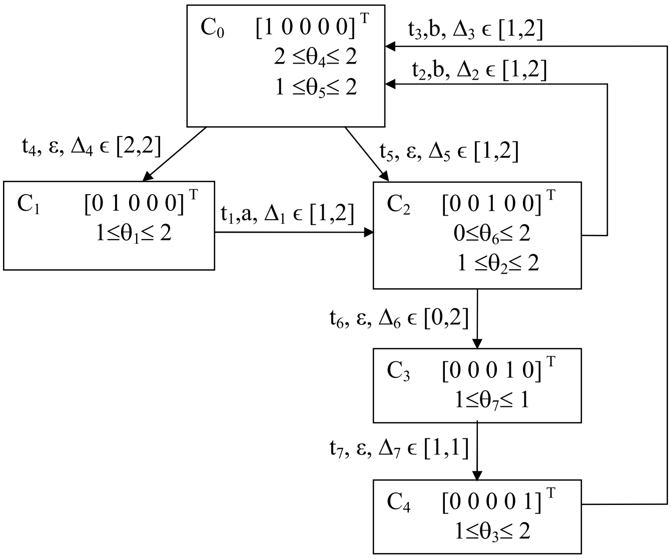

Fig. 2. The MSCG relative to the TPN in Fig. 1.

If the set of constraints (1) is feasible, let $DiagK_i = 0$: It means that there exist two sequences as the ones described in the itemized list of Definition 1, then the system is not K-diagnosable w.r.t. fault class i. ■

***Proposition** 7:* Given a labeled TPN system $\langle N_d, M_0 \rangle$ with labeling function $\mathcal{L}$, fault classes T_f^i, $i \in \{1, \ldots, r\}$, and an integer number K, Algorithm 6 allows to establish if the system is/is not K-diagnosable w.r.t. fault class T_f^i, $i \in \{1, \ldots, r\}$. In particular, if $DiagK_i = 1$ then the system is K diagnosable w.r.t. fault class i, if $DiagK_i = 0$ then the system is not K diagnosable w.r.t. fault class i.

Proof: At Step 1 we build the MSCG since it collects all the information on the evolution of the TPN system. By Assumptions A1 and A2 the number of classes of the MSCG is finite [4]. Moreover we let $DiagK_i = 1$, for all $i \in \{1, \ldots, r\}$, namely we initially assume that the system is K-diagnosable w.r.t. all fault classes.

At Step 2 we compute the logic version of the TPN system by considering the graph G isomorphic to the MSCG. If the logic system is K-diagnosable, all the more reason it is K-diagnosable the TPN system (see Proposition 2), so flag variables $DiagK_i$ are not changed. To the best of our knowledge all the approaches in the literature that allow to perform such an analysis are based on Assumption A3. This is the reason why such an assumption is actually made as one of the main assumptions of the proposed approach.

If the logic system is not K-diagnosable, we need to verify if the sequences that are "ambiguous" in the underlying logic PN system, are also ambiguous in the TPN system, i.e., they produce the same observations in the same time instants. This verification is done by checking the feasibility of constraints (1) whose physical meaning is explained in the following. Obviously, this needs to be done for each pair of paths corresponding to the pair of sequences σ', σ''.

— Constraints (a) impose limitations on the total length of the sequence. In particular, (a1) (resp. (a3)) implies that, if the sequence really corresponds to the considered path, then the class at the end of the path, namely $C_{q_{k'}}$ (resp. $C_{q_{k''}}$), is reached at a time that is smaller than or equal to the time instant $\bar{\tau}$. Note that the time instant $\bar{\tau}$ is an unknown but should be equal for the two sequences. Now, since Δ'_l (resp. Δ''_l) denotes the time interval to go from $C_{q_{l'-1}}$ to $C_{q_{l'}}$ (resp. from $C_{q_{l''-1}}$ to $C_{q_{l''}}$), the sum in the left hand side of (a1) (resp. (a3)) is equal to the time spent to reach class $C_{q_{k'}}$ (resp. $C_{q_{k''}}$). Constraint (a2) (resp. (a4)) imposes that class $C_{q_{k'}}$ (resp. $C_{q_{k''}}$) is actually the last visited class. This is verified if the difference between $\bar{\tau}$ and the time spent to reach class $C_{q_{k'}}$ (resp. $C_{q_{k''}}$) is less than the minimum of the upper bounds of the output transitions of class $C_{q_{k'}}$ (resp. $C_{q_{k''}}$). Indeed, once the node $C_{q_{k'}}$ (resp. $C_{q_{k''}}$) is reached a transition enabled at $M_{q_{k'}}$ (resp. $M_{q_{k''}}$) will fire at most after a time interval equal to the minimum of the upper bounds of the transitions enabled at $C_{q_{k'}}$ (resp. $C_{q_{k''}}$).

— Constraints (b) simply impose the limitations on the time intervals Δ'_l ((b1) and (b2)) and Δ''_l ((b3) and (b4)) as given in the MSCG.

— Constraints (c) imply that the observable transitions of the two sequences σ' and σ'' occur at the same time instant.

— Constraint (d) imposes that $\bar{\tau}$ is a nonnegative real number.

If constraints (1) are feasible for a class i, then the corresponding flag variable $DiagK_i$ is set to 0 because the system is not K-diagnosable w.r.t. that fault class. □

Note that constraints (1) are clearly nonlinear. However, they can easily be linearized using the procedure in [4]. For the sake of brevity, the linear constraints corresponding to (1) are not reported here.

Some remarks should be done concerning Step 3 of Algorithm 6.

Several approaches have been proposed in the DES literature to compute pairs of sequences with the same observable projection, one containing the fault and the other one not. Two are the most commonly used. The first one has been proposed by Sampath *et al.* in [24] and is based on the notion of *diagnoser*. The second one has been proposed by Yoo and Lafortune [29] and uses the notion of *verifier* automaton. In simple words the verifier automaton is the parallel composition of the system with faults and the system without faults, with a synchronization on the observable events.

In this section we present a numerical example to illustrate Algorithm 6. Such an example has been implemented using the approach in [24] to compute ambiguous sequences. This choice has been done, even if the technique in [29] is more efficient from a computational point of view as discussed in Section VII, for the following three main reasons. First, it is probably more intuitive. Second, it allows to deal with more than one fault class simultaneously, while a different verifier should be constructed for each fault class. Finally, an efficient software [27], called UMDES, can be used to compute the diagnoser and the cycles.

In the following subsection we discuss how Algorithm 6 should be particularized when using the technique in [24].

A. Analysis with the diagnoser approach

When the technique in [24] is used to compute ambiguous sequences, Steps 3 and 4 of Algorithm 6 should be rewritten as follows.

3') Compute the diagnoser $Diag(G)$ using the approach in [24] and look for uncertain states[1] for any fault class $i \in \{1, \ldots, r\}$.

4') *If* there are no uncertain states for any fault class $i \in \{1, \ldots, r\}$, *then* the system is K-diagnosable (this holds for any $K \in \mathbb{N}$).
Else if there are uncertain paths[2] for some fault class $i \in \{1, \ldots, r\}$ (that may eventually include cycles), *then* consider all pairs of sequences in G associated with the corresponding uncertain path such that in the faulty sequence the number of observable events after the fault is K, and go to Step 5.

Steps 3' and 4' can be explained as follows.

From [24] we know that if the diagnoser $Diag(G)$ does not contain uncertain states for any fault class T_f^i, then G is K-diagnosable for any $K \in \mathbb{N}$. Thus by Proposition 2 the TPN system $\langle N_d, M_0 \rangle$ is also K-diagnosable for any $K \in \mathbb{N}$.

If there are uncertain paths including uncertain but not indeterminate cycles[3] in $Diag(G)$ for some fault class T_f^i, then for any of such classes there exists a $K^*(i) \in \mathbb{N}$ such that the underling logic system is $K^*(i)$-diagnosable w.r.t. to T_f^i. Thus by Proposition 2 there exists a $\bar{K}(i) \leq K^*(i)$ such that the TPN system $\langle N_d, M_0 \rangle$ is $\bar{K}(i)$-diagnosable w.r.t. T_f^i. However, since we are not able to evaluate $\bar{K}(i)$ (as well as $K^*(i)$), we have to consider all pairs of sequences in G associated with the corresponding uncertain path and such that the number of observable events after the fault is K (Step 5).

All the more reason, pairs of sequences in G associated with uncertain paths should be considered in Step 5 if the uncertain paths contain indeterminate cycles.

B. A numerical example

Let us consider again the TPN system and its MSCG shown respectively in Figures 1 and 2, and introduced in Example 5. We want to know if such a system is 2-diagnosable.

The nondeterministic automaton G, isomorphic to the MSCG in Fig. 2, is shown in Fig. 3, and its diagnoser $Diag(G)$, obtained using the approach in [24], is illustrated in Fig. 4.

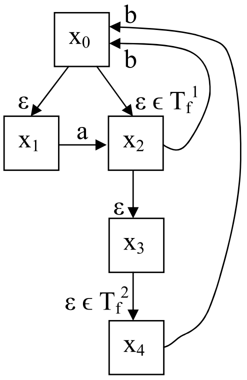

Fig. 3. The nondeterministic automaton G isomorphic to the MSCG of Fig. 1.

Each state of the diagnoser is composed by a set of states of the automaton G and by a label that indicates if reaching a certain state x_i from x_0 no fault has occurred (N, that stands for $N1N2$), or fault transition $t_5 \in T_f^1$ has fired but not $t_7 \in T_f^2$ ($F1$, that stands for $N2F1$), or, viceversa, fault transition t_7 has fired but not t_5 ($F2$, that stands for $N1F2$), or both fault transitions t_5 and t_7 have occurred ($F1F2$). Note that the same state can have different labels because there could be more paths from state x_0 to state x_i.

Looking at G we see that there are three uncertain states w.r.t. T_f^1, namely S_1, S_4 and S_8, and no uncertain path (with more than one state) or cycle. This implies that the TPN is K-diagnosable w.r.t. the first fault class for any $K \geq 1$. Indeed as soon as a new observation occurs we can establish if a fault in the first class has either occurred or not, since any further observation leads to a certain state.

On the contrary, there are three elementary uncertain cycles w.r.t. the second fault class, namely, S_3bS_3, $S_5bS_7aS_5$, $S_6bS_8aS_6$, and $S_3aS_5bS_7bS_3$. In particular, three out of four uncertain cycles for the second fault class are also indeterminate: S_3bS_3, $S_5bS_7aS_5$, $S_6bS_8aS_6$.

As an example, consider the uncertain cycle $S_6bS_8aS_6$. Let $\sigma' = (t_4, \tau_1')\ (t_1, \tau_2')\ (t_2, \tau_3')\ (t_4, \tau_4')\ (t_1, \tau_5')\ (t_2, \tau_6')\ (t_4, \tau_7')\ (t_1, \tau_8')$ and $\sigma'' = (t_4, \tau_1'')\ (t_1, \tau_2'')\ (t_2, \tau_3'')\ (t_4, \tau_4'')\ (t_1, \tau_5'')\ (t_6, \tau_6'')\ (t_7, \tau_7'')\ (t_3, \tau_8'')\ (t_4, \tau_9'')\ (t_1, \tau_{10}'')$ be two timed sequences in the MSCG associated with the pair of sequences $x_0 \varepsilon x_1 a x_2 b x_0 \varepsilon x_1 a x_2 b x_0 \varepsilon x_1 a$ and $x_0 \varepsilon x_1 a x_2 b x_0 \varepsilon x_1 a x_2 \varepsilon x_3 \varepsilon x_4 b x_0 \varepsilon x_1 a$ in G associated with the indeterminate cycle $S_6bS_8aS_6$ in the diagnoser. Sequence σ'' includes $t_7 \in T_f^2$ and such a transition is followed by 2 observable transitions, precisely t_3 and t_1. Transition t_7 is not in σ'.

According to the set of constraints (1) we associate the following set of constraints to σ' and σ'':

[1] A diagnoser state is *uncertain* w.r.t. a fault class i if it has at least a system state with label Ni (no fault) for that class and at least one state with label Fi (fault).

[2] A *path* (*cycle*) in $Diag(G)$ is *uncertain* w.r.t. a fault class i if it is composed by uncertain states w.r.t. a fault class i.

[3] A cycle in $Diag(G)$ is *indeterminate* w.r.t. fault class i if it is uncertain and if there exist: (i) a corresponding cycle (of observable events) in G involving only states that carry Fi in their labels in the cycle in $Diag(G)$, and (ii) a corresponding cycle (of observable events) in G involving only states that carry Ni in their labels in the cycle in $Diag(G)$. In [24] a procedure to determine if an uncertain cycle is indeterminate is presented.

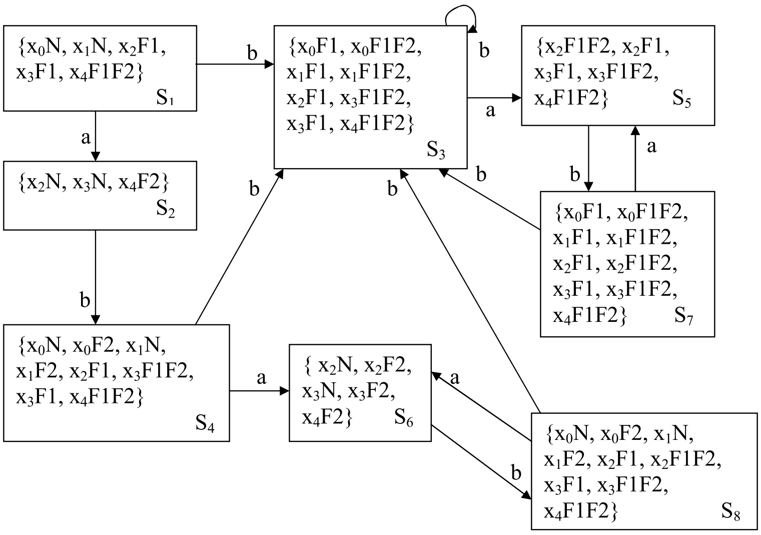
{x0N, x1N, x2F1, x3F1, x4F1F2} S1
{x2N, x3N, x4F2} S2
{x0F1, x0F1F2, x1F1, x1F1F2, x2F1, x3F1F2, x3F1, x4F1F2} S3
{x0N, x0F2, x1N, x1F2, x2F1, x3F1F2, x3F1, x4F1F2} S4
{x2F1F2, x2F1, x3F1, x3F1F2, x4F1F2} S5
{ x2N, x2F2, x3N, x3F2, x4F2} S6
{x0F1, x0F1F2, x1F1, x1F1F2, x2F1, x2F1F2, x3F1, x3F1F2, x4F1F2} S7
{x0N, x0F2, x1N, x1F2, x2F1, x2F1F2, x3F1, x3F1F2, x4F1F2} S8
a
b
b
b
a
b
b
b
b
a
a
a
b

Fig. 4. The diagnoser of automaton G in Fig. 3.

$$\left\{\begin{array}{l}
\left.\begin{array}{l}
\Delta'_1+\Delta'_2+\Delta'_3+\Delta'_4+\Delta'_5+\Delta'_6+ \\
\qquad\qquad +\Delta'_7+\Delta'_8 \leq \bar{\tau} \\
\bar{\tau}-(\Delta'_1+\Delta'_2+\Delta'_3+\Delta'_4+ \\
\qquad\qquad +\Delta'_5+\Delta'_6+\Delta'_7+\Delta'_8)<2 \\
\Delta''_1+\Delta''_2+\Delta''_3+\Delta''_4+\Delta''_5+\Delta''_6+\Delta''_7+ \\
\qquad +\Delta''_8+\Delta''_9+\Delta''_{10} \leq \bar{\tau} \\
\bar{\tau}-(\Delta''_1+\Delta''_2+\Delta''_3+\Delta''_4+\Delta''_5+ \\
\qquad \Delta''_6+\Delta''_7+\Delta''_8+\Delta''_9+\Delta''_{10})<2
\end{array}\right\}(a) \\
\left.\begin{array}{llll}
\Delta'_1 \geq 2, & \Delta'_1 \leq 2, & \Delta'_2 \geq 1, & \Delta'_2 \leq 2 \\
\Delta'_3 \geq 1, & \Delta'_3 \geq 2, & \Delta'_4 \leq 2, & \Delta'_4 \leq 2 \\
\ldots & & & \\
\ldots & & & \\
\Delta''_1 \geq 2, & \Delta''_1 \leq 2, & \Delta''_2 \geq 1, & \Delta''_2 \leq 2 \\
\Delta''_3 \geq 1, & \Delta''_3 \leq 2, & \ldots, & \ldots
\end{array}\right\}(b) \\
\left.\begin{array}{l}
\Delta'_1+\Delta'_2=\Delta''_1+\Delta''_2 \\
\Delta'_1+\Delta'_2+\Delta'_3=\Delta''_1+\Delta''_2+\Delta''_3 \\
\ldots \\
\Delta'_1+\Delta'_2+\Delta'_3+\Delta'_4+\Delta'_5+\Delta'_6= \\
\qquad =\Delta''_1+\Delta''_2+\Delta''_3+\Delta''_4+\Delta''_5+ \\
\qquad\qquad +\Delta''_6+\Delta''_7+\Delta''_8 \\
\ldots
\end{array}\right\}(c) \\
\bar{\tau} \geq 0 \qquad (d)
\end{array}\right.$$

In the above set of constraints Δ'_1 denotes the time interval spent in class C_0. At time $\tau'_1 = \Delta'_1$ transition t_4 fires leading to class C_1. Δ'_2 denotes the time interval spent in class C_1. At time $\tau'_2 = \Delta'_1 + \Delta'_2$ transition t_1 fires leading to class C_2, and so on. Note that Δ'_4 denotes the time interval spent in class C_1 the second time this class is visited.

The existence of a solution has been tested using the LINDO package[4], introducing a dummy objective function (we used $\min \bar{\tau}$) to obtain a LPP formulation. In particular, we found as optimal solution $\bar{\tau} = 12$ that corresponds to $\mathcal{L}(\sigma') = \mathcal{L}(\sigma'') = (a, 3)\ (b, 4)\ (a, 7)\ (b, 9)\ (a, 12)$. Therefore the system is not 2-diagnosable with respect to the second fault class.

We conclude this section observing that K-diagnosability strictly depends on transitions timing. As an example, no solution is found if we assume $Q(t_6) = (1, 3)$ instead of $Q(t_6) = (0, 2)$.

VI. τ-DIAGNOSABILITY ANALYSIS

In this section we propose a procedure to analyze the τ-diagnosability of a TPN. Main steps are summarized by the following algorithm.

***Algorithm** 8:* [Analysis of τ-diagnosability]

Inputs: A labeled TPN system $\langle N_d, M_0 \rangle$ with labeling function $\mathcal{L}$, fault classes T_f^i, $i \in \{1, \ldots, r\}$ and a real positive number τ.

Output: $Diag\tau_i \in \{0, 1\}$, $i \in \{1, \ldots, r\}$: $Diag\tau_i = 1$ ($Diag\tau_i = 0$) means that the system is (is not) τ-diagnosable w.r.t. fault class T_f^i.

1) Build the MSCG according to Algorithm 1 and let $Diag\tau_i = 1$ for all $i \in \{1, \ldots, r\}$.
2) Compute the nondeterministic automaton G isomorphic to the MSCG. Each state of G corresponds to a class of the MSCG. Edges are labeled with the second element of the edges of the MSCG, namely the label associated with transitions. In the case of silent transitions modeling a fault, it is also specified to which fault class transitions belong.
3) For any fault class $i \in \{1, \ldots, r\}$ compute all pairs of sequences in G having the same observable projection, one containing a fault transition in T_f^i and the other one not, and such that in the faulty sequence, the subsequence that fires after the fault has the sum of the path's transitions lower bounds less than or equal to τ and the sum of the path's transitions upper bounds greater than or equal to τ.

[4] LINDO website: http://www.lindo.com/

4) *If* there are no such sequences for any fault class $i \in \{1,\ldots,r\}$, *then* the system is τ-diagnosable.
   Else if for some fault class $i \in \{1,\ldots,r\}$ Step 3 provides some pair, *then* consider all such pairs and go to Step 5.
5) Given a pair of sequences in G and a fault class T_f^i, find the corresponding timed sequences $\sigma' = (t_{i'_1},\tau'_1)\ldots(t_{i'_{k'}},\tau'_{k'})$ and $\sigma'' = (t_{i''_1},\tau''_1)\ldots(t_{i''_{k''}},\tau''_{k''})$ in the MSCG. Let π' and π'' be defined as in Step 3 of Algorithm 6, namely, respectively, the paths associated with sequences σ' and σ'' (containing the fault) in the MSCG.
   Let $\bar{k} < k''$ be the index associated with the first occurrence of the fault transition in σ''. We associate with the two paths π' and π'' the following set of constraints:

$$\left\{\begin{array}{ll}
\sum_{l=1}^{k'} \Delta'_l \leq \bar{\tau}, & (a1)\\
\bar{\tau} - \sum_{l=1}^{k'} \Delta'_l < \min_{r:t_r \in out(C_{q_{k'}})} \{u_r^{q_{k'}}\}, & (a2)\\
\sum_{l=1}^{k''} \Delta''_l \leq \bar{\tau}, & (a3)\\
\bar{\tau} - \sum_{l=1}^{k''} \Delta''_l < \min_{r:t_r \in out(C_{q_{k''}})} \{u_r^{q_{k''}}\}, & (a4)\\
\bar{\tau} - \sum_{l=1}^{\bar{k}} \Delta''_l = \tau & (a5)\\
\Delta'_l \geq \max\{0, l_{i_l}^{q_{l-1}}\}, & l = 1,\ldots,k', \ (b1)\\
\Delta'_l \leq \min_{j:t_j \in \mathcal{A}(M_{q_{l-1}})} \{u_j^{q_{l-1}}\}, & l = 1,\ldots,k', \ (b2)\\
\Delta''_l \geq \max\{0, l_{i_l}^{q_{l-1}}\}, & l = 1,\ldots,k'', \ (b3)\\
\Delta''_l \leq \min_{j:t_j \in \mathcal{A}(M_{q_{l-1}})} \{u_j^{q_{l-1}}\}, & l = 1,\ldots,k'', \ (b4)\\
\sum_{l=1}^{q'} \Delta'_l = \sum_{l=1}^{q''} \Delta''_l, & \forall t_{i'_{q'}}, t_{i''_{q''}} \in T_o \text{ with same label and same observable prefix} \ (c)\\
\bar{\tau} \geq \tau & (d)
\end{array}\right. \tag{2}$$

   If the set of constraints (2) is feasible, let $Diag\tau_i = 0$: It means that there exist two sequences as the ones described in the itemized list of Definition 3, *then* the system is not τ-diagnosable w.r.t. fault class i. ■

Proposition 9: Given a labeled TPN system $\langle N_d, M_0\rangle$ with labeling function $\mathcal{L}$, fault classes T_f^i, $i \in \{1,\ldots,r\}$, and a real positive number τ, Algorithm 8 allows to establish if the system is/is not τ-diagnosable w.r.t. fault class T_f^i, $i \in \{1,\ldots,r\}$. In particular, if $Diag\tau_i = 1$ then the system is τ diagnosable w.r.t. fault class i, if $Diag\tau_i = 0$ then the system is not τ diagnosable w.r.t. fault class i.

Proof: As in Algorithm 6, at Step 1 we compute the MSCG since it collects all the information on the evolution of the TPN system. By Assumptions A1 and A2 we know for sure that such a graph is finite [4]. We consider the logic version of our PN system by considering the graph G isomorphic to it (Step 2).

We consider all pairs of sequences that are ambiguous in the underlying logic model and such that the time length of the subsequence after the fault is potentially equal to τ. In more detail (see Step 3) if τ is less than the sum of the path's transitions lower bounds then such sequence can never fire in an amount of time equal to τ, so we should not take it into account. The same holds if τ is greater than the sum of the path's transitions upper bounds. Note that Assumption A3' allows us to have a stop criterion. In fact, if Assumption A3' is not verified we could have cycles whose sum of the lower bounds of all transitions is zero. If such is the case the condition "τ greater than or equal to the sum of the minimum lower bounds of the subsequence firing after the fault" is always verified and we should consider sequences of infinite length.

If no such pair exists, then the system is τ-diagnosable, else we have to check if some of such pairs are actually enabled in the TPN system, i.e., if there exist two sequences that can produce the same observations in the same time instants. This verification is done by solving the set of constraints (2) that differs from (1) in constraint (d) and also includes constraint (a5). In particular, constraint (a5) guarantees that the duration in time of σ'' after the first occurrence of a fault transition in T_f^i is τ (third item of Definition 3). Finally, constraint (d) guarantees the feasability of the constraints, in fact if such a constraints is not verified, constraint (a5) is never satisfied. □

In the following we present a numerical example to illustrate the approach. As in the case of K-diagnosabily we use the diagnoser to compute ambiguous strings in the underling logic model. Details on this are provided in the following subsection.

A. Analysis with the diagnoser approach

When the diagnoser approach is used to compute ambiguous strings, Steps 3 and 4 of Algorithm 8 should be replaced by the following Steps 3" and 4".

3") Compute the diagnoser $Diag(G)$ using the approach in [24] and look for uncertain states for any fault class $i \in \{1,\ldots,r\}$.

4") *If* there are no uncertain states for any fault class $i \in \{1,\ldots,r\}$, *then* the timed system is τ-diagnosable (this holds for any $\tau \in \mathbb{R}^+$).

Else if there are *single* uncertain states w.r.t. some fault class $i \in \{1, \ldots, r\}$, *then* for each uncertain state consider the corresponding states in G. Consider all exiting edges from such states and take the minimum among such upper bounds associated with the transitions that label the exiting edges. If τ is greater than or equal to the minimum among such upper bounds, then the system can be τ-diagnosable, (but we still have to look for uncertain paths, see next *Else if*), *else* the system is not τ-diagnosable w.r.t. fault class i and we let $Diag\tau_i = 0$.

Else if there are uncertain paths (including at least two states and eventually cycles) for some fault class $i \in \{1, \ldots, r\}$, *then* consider all pairs of sequences in G associated with the corresponding uncertain path such that in the faulty sequence the subsequence that fires after the fault has the sum of the path's transitions lower bounds less than or equal to τ, and the sum of the path's transitions upper bounds greater than or equal to τ, and go to Step 5.

The above two steps can be explained as follows.

From [24] we know that if the diagnoser $Diag(G)$ does not contain uncertain states for any fault class T_f^i, then G is diagnosable. Thus, by Proposition 4 the TPN system $\langle N_d, M_0\rangle$ is also τ-diagnosable for any $\tau \in \mathbb{R}^+$.

The first *Else if* follows from the fact that if for some fault class T_f^i, with $i \in \{1, \ldots, r\}$, there exists a single uncertain state in $Diag(G)$ such that τ is less than the minimum among the upper bounds associated with the transitions that label the exiting edges of the uncertain state, then the TPN system $\langle N_d, M_0\rangle$ is not τ-diagnosable w.r.t. class T_f^i. Indeed, if such a case occurs, then τ time units after the fault has occurred, the system is still in an ambiguous situation, then the TPN system $\langle N_d, M_0\rangle$ is not τ-diagnosable w.r.t. the ith class. If such is not the case, we have to look for uncertain paths (second *Else if*).

The second *Else if* follows from the fact that, by Definition 3, if for some fault class T_f^i there exist an uncertain path in $Diag(G)$, and two sequences σ' and σ'' in the MSCG that satisfy the following three items:

- σ' and σ'' satisfy the first two items of Definition 1,
- the subsequence of σ'' that fires after the first fault in T_f^i has the sum of the path's transitions lower bounds less than or equal to τ and the sum of path's transitions upper bounds greater than or equal to τ,
- σ' and σ'' lead to a feasible solution for constraints in eq. (2),

then the TPN system $\langle N_d, M_0\rangle$ is not τ-diagnosable.

B. A numerical example

Let us consider again the TPN system in Example 5 and Subsection V-B where K-diagnobility has been studied in the case of $K = 2$.

Now, we want to know if the system is τ-diagnosable w.r.t. the second fault class for $\tau = 3$.

Consider the uncertain path $S_6 b S_8 a S_6$ in the diagnoser. Let $\sigma' = (t_4, \tau_1')$ (t_1, τ_2') (t_2, τ_3') (t_4, τ_4') (t_1, τ_5') $(t_2, \tau_6'))$ and $\sigma'' = (t_4, \tau_1'')$ (t_1, τ_2'') (t_2, τ_3'') (t_4, τ_4'') (t_1, τ_5'') (t_6, τ_6'') (t_7, τ_7'') (t_3, τ_8'') be two timed sequences in the MSCG associated with the pair of sequences $x_0 \varepsilon x_1 a x_2 b x_0 \varepsilon x_1 a x_2 b$ and $x_0 \varepsilon x_1 a x_2 b x_0 \varepsilon x_1 a x_2 \varepsilon x_3 \varepsilon x_4 b$ in G relative to the uncertain path at hand. Only sequence σ'' includes $t_7 \in T_f^2$ and it is followed by one observable transition (t_3). Moreover, the subsequence $t_7 t_3$ is such that the sum of the path's transitions lower bounds is less than or equal to 2 and the sum of path's transitions upper bounds is greater than or equal to 3 (in particular, the sum of lower bounds is equal to 2 and the sum of the upper bounds is equal to 3).

According to the set of inequalities in (2), we associate the following set of constraints with these two sequences:

$$
\left\{
\begin{array}{l}
\left.
\begin{array}{l}
\Delta_1' + \Delta_2' + \Delta_3' + \Delta_4' + \Delta_5' + \Delta_6' \leq \bar{\tau} \\
\bar{\tau} - (\Delta_1' + \Delta_2' + \Delta_3' + \Delta_4' + \\
\qquad\qquad + \Delta_5' + \Delta_6') < 2 \\
\Delta_1'' + \Delta_2'' + \Delta_3'' + \Delta_4'' + \Delta_5'' + \Delta_6'' + \Delta_7'' + \\
\qquad + \Delta_8'' \leq \bar{\tau} \\
\bar{\tau} - (\Delta_1'' + \Delta_2'' + \Delta_3'' + \Delta_4'' + \Delta_5'' + \Delta_6'' + \Delta_7'' + \\
\qquad + \Delta_8'') < 2 \\
\bar{\tau} - (\Delta_1'' + \Delta_2'' + \Delta_3'' + \Delta_4'' + \Delta_5'' + \Delta_6'') = 3
\end{array}
\right\} (a) \\
\left.
\begin{array}{l}
\Delta_1' \geq 2, \quad \Delta_1' \leq 2, \quad \Delta_2' \geq 1, \quad \Delta_2' \leq 2 \\
\Delta_3' \geq 1, \quad \Delta_3' \geq 2, \quad \Delta_4' \leq 2, \quad \Delta_4' \leq 2 \\
\ldots \\
\ldots \\
\Delta_1'' \geq 2, \quad \Delta_1'' \leq 2, \quad \Delta_2'' \geq 1, \quad \Delta_2'' \leq 2 \\
\Delta_3'' \geq 1, \quad \Delta_3'' \leq 2, \quad \ldots \\
\ldots
\end{array}
\right\} (b) \\
\left.
\begin{array}{l}
\Delta_1' + \Delta_2' = \Delta_1'' + \Delta_2'' \\
\Delta_1' + \Delta_2' + \Delta_3' = \Delta_1'' + \Delta_2'' + \Delta_3'' \\
\ldots \\
\Delta_1' + \Delta_2' + \Delta_3' + \Delta_4' + \Delta_5' + \Delta_6' = \\
\qquad = \Delta_1'' + \Delta_2'' + \Delta_3'' + \Delta_4'' + \Delta_5'' + \\
\qquad\qquad + \Delta_6'' + \Delta_7'' + \Delta_8''
\end{array}
\right\} (c) \\
\bar{\tau} \geq 0 \qquad (d)
\end{array}
\right.
$$

Using the LINDO package we find out that such a set of constraints is feasible. In particular, assuming again a performance index to be minimized equal to $\bar{\tau}$, we found the optimal solution $\bar{\tau} = 10$ that corresponds to $\mathcal{L}(\sigma') = \mathcal{L}(\sigma'') = (a, 3)$ $(b, 4)$ $(a, 7)$ $(b, 9)$. Therefore the system is not diagnosable in 3 time units w.r.t. the second fault class.

VII. Complexity of the procedure

In this section we analyze the complexity of each step of the procedure to analyze K-diagnosability. Similar results can be found if we study the computational complexity to analyze the τ-diagnosability. They are not reported here for the sake of brevity.

1) *Construction of the MSCG.* The number of nodes of the MSCG increases exponentially with the system complexity (net structure, and number of tokens in the initial marking).
2) *Computation of the nondeterministic automaton G isomorphic to the MSCG.* It is obtained from the MSCG simply working on the labels of the edges of the MSCG, so requires no additional computational effort.

3) *Computation of the pairs of ambiguous sequence in the underlying logic model.*
As already discussed, this step depends on the approach considered, e.g. [24] or [29].
In the first case, if n_G denotes the number of nodes of the nondeterministic automaton G isomorphic to the MSCG, the size of $Diag(G)$ is 2^{n_G}, i.e., it is exponential in the number of states of G. Then uncertain paths including indeterminate cycles for some fault class T_f^i should be computed. This can be obtained in polynomial time in the cardinality of the state space of the diagnoser $Diag(G)$.
In the second case [29] the number of states of the verifier automaton grows polynomially with the number n_G of states of G, and the complexity of determining the ambiguous sequences is also polynomial w.r.t. n_G.

4) *Verification of the ambiguity of a given pair of sequences wrt time.*
For any pair of sequences that is ambiguous in the underlying logic model wrt a given fault class (such that the number of observable transitions after the fault is K), we first determine the corresponding pairs of timed sequences $\sigma' = (t_{i'_1}, \tau'_1) \ldots (t_{i'_{k'}}, \tau'_{k'})$ and $\sigma'' = (t_{i''_1}, \tau''_1) \ldots (t_{i''_{k''}}, \tau''_{k''})$ in the MSCG. Let l_{max} be the length of the longest of such sequences. To establish if such sequences are ambiguous also in the TPN model, we associate with them a set of non linear constraints of the form (1), and evaluate if they are feasible. This can be done solving an LPP obtained linearizing the constraints (a1) and (b1) (see [4] for the details on linearization of such constraints). This LPP has a number of constraints at most equal to $2 + 2(|T| + 2l_{max}|T|) + l_{max}+1$. In particular, one constraint is of type (a1) and one of type (a3). The number of constraints of type (a2) and (a4) is equal to the number of output transitions of the last node in the path associated with each sequence, that is at most equal to $|T|$. Constraints of type (b1), (b2), (b3) and (b4) are each in a number equal at most to $l_{max}|T|$. The number of constraints of type (c) is equal to the number of observable transitions that cannot be greater than l_{max}. The number of constraints of type (d) is one. Finally, the number of unknowns of each LPP is equal to $k' + k'' + 1$, namely to the length of the two sequences plus one ($\bar{\tau}$), that is at most equal to $2l_{max} + 1$.

We finally note that our approach is based on the solution of some LPPs that can be solved using polynomial time algorithms. Moreover there are many softwares, such as CPLEX, that allow to compute LPPs in a very efficient way. However, if the system evolution is very fast compared with the time required to solve the LPPs, it may be impossible to use our approach. Obviously, this is a limitation of the procedure, but it is common to all approaches that require online computations.

VIII. Conclusions and future work

In this paper we have analyzed K-diagnosability and τ-diagnosability of labeled TPN systems. We have first formalized the notions of K-diagnosability and τ-diagnosability for a TPN. Then, using the Modified State Class Graph, we have presented two procedures to analyze K-diagnosability and τ-diagnosability of a labeled time Petri net system, respectively. Both approaches require a preliminary diagnosability analysis of the underlying logic model, and the solution of some linear programming problems.

Several are the directions of our future research in this area. First, we want to validate the effectiveness of the proposed approach on a real case study, e.g. in the manufacturing or transportation domain. Second, we plan to investigate the possibility of making K-diagnosable (τ-diagnosable) a TPN system that is not K-diagnosable (τ-diagnosable), acting on the time intervals associated with transitions. Finally, we plan to extend the proposed results to a decentralized/distributed framework.

References

[1] D. Corona A. Giua and C. Seatzu. State estimation of λ-free labeled Petri nets with contact-free nondeterministic transitions. *Discrete Event Dynamic Systems*, 15(1):85–1, 2005.

[2] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183 – 235, 1994.

[3] F. Basile, M.P. Cabasino, and C. Seatzu. K-diagnosability of Time labeled Petri nets. In *Proc. IFAC WODES'14: 12th Work. on Discrete Event Systems*, 2014.

[4] F. Basile, M.P. Cabasino, and C. Seatzu. State estimation and fault diagnosis of Time labeled Petri net systems with unobservable transitions. *IEEE Trans. on Automatic Control*, 2014.

[5] F. Basile, P. Chiacchio, and D. Teta. A hybrid model for real time simulation of urban traffic. *Control Engineering Practice*, 20(2):123–137, 2012.

[6] 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):748–759, 2009.

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

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

[9] B. Bérard, F. Cassez, S. Haddad, D. Lime, and O.H. Roux. Comparison of Different Semantics for Time Petri Nets. In *Proc. 3^{rd} international conference on Automated technology for verification and analysis*, 2005.

[10] Patrice Bonhomme. State Observer Synthesis of Real-Time Systems Modeled by P-Time Petri Nets. In 18^{th} *IEEE Int. Conf. on Emerging Technologies & Factory Automation*, 2013.

[11] B.A. Brandin and W.M. Wonham. Supervisory control of timed discrete-event systems. *IEEE Transactions on Automatic Control*, 39(2):329–342, 1994.

[12] M. P. Cabasino, A. Giua, and C. Seatzu. Fault detection for discrete event systems using Petri nets with unobservable transitions. *Automatica*. Submitted.

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

[14] 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, 2011.

[15] M.P. Cabasino, A. Giua, and C. Seatzu. Diagnosability of discrete-event systems using labeled Petri nets. *IEEE Trans. on Automation Science and Engineering*, 11:144 – 153, 2014.

[16] F. Cassez. Dynamic observers for fault diagnosis of timed systems. In *Proc. 49th IEEE Conf. on Decision and Control*, 2010.

[17] M. Dotoli, M.P. Fanti, and A.M. Mangini. Fault detection of discrete event systems using Petri nets and integer linear programming. In *Proc. of 17th IFAC World Congress*, Seoul, Korea, July 2008.

[18] S. Hashtrudi Zad, R.H. Kwong, and W.M. Wonham. Fault diagnosis in discrete-event systems: incorporating timing information. *IEEE Transactions on Automatic Control*, 50(7):1010–1015, 2005.

[19] G. Jiroveanu and R.K. Boel. A distributed approach for fault detection and diagnosis based on Time Petri Nets. *Mathematics and Computers in Simulation*, 70(5-6):287 – 313, 2006.

[20] Philip Meir Merlin. *A study of the recoverability of computing systems.* PhD thesis, Univ. of California, Irvine, 1974.

[21] T. Murata. Petri nets: properties, analysis and applications. *Proc. of the IEEE*, 77(4):541–580, 1989.

[22] Yannick Pencolé and Audine Subias. A Chronicle-based Diagnosability Approach for Discrete Timed-event Systems: Application to Web-Services. *Journal of Universal Computer Science*, 15(17):3246–3272, 2009.

[23] C. Ramchandani. Analysis of asynchronous concurrent systems by timed Petri nets. Technical report, Massachusetts Inst. of Technology, Cambridge, MA, USA, 1974.

[24] 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, 1995.

[25] C. Seatzu, M. Silva, and J.H. van Schuppen (Eds). *Control of Discrete-Event Systems. Automata and Petri Net Perspectives*, volume 433. in Lecture Notes in Control and Information Science, Springer, 2012.

[26] S. Tripakis. Fault Diagnosis for Timed Automata. *Lecture Notes in Computer Science*, 2469:205–221, 2002.

[27] Univ. of Michigan. Executables of the umdes-lib software library for solaris, linux, mac and windows are publicly available. *http://www.eecs.umich.edu/umdes/toolboxes.html*. 2007.

[28] X. Wang, C. Mahulea, and M. Silva. Fault Diagnosis Graph of Time Petri Nets. In *2013 European Control Conference (ECC'13), Zurich, Switzerland*, 2013.

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

[30] J. Zaytoon and S. Lafortune. Overview of fault diagnosis methods for Discrete Event Systems. *Annual Reviews in Control*, 37:308 – 320, 2013.

Francesco Basile Francesco Basile was born in Naples, Italy, in 1971. He received the Laurea degree in Electronic Engineering in 1995 and the Ph.D. degree in Electronic and Computer Engineering in 1999 from the University of Naples. In 1999 he was visiting researcher for six months at the Departamento de Ingenieria Informatica y Systemas of the University of Saragoza, Spain. He is currently associate professor of Automatic Control at the Dipartimento di Ingegneria Elettronica ed Ingegneria Informatica of the University of Salerno, Italy. He has published more than 100 papers on international journals and conferences He has been member of the editorial board of International Journal of Robotics and Automation. He is a member of the editorial board of IEEE Transactions on Control Systems Technology, IEEE Transactions on Automation Science and Engineering and of IEEE Control System Society Conference Editorial Board. His current research interest are: modelling and control of discrete event systems, automated manufacturing and robotic. He is IEEE Senior member since November 2011.

Maria Paola Cabasino Maria Paola Cabasino received the Laurea degree in electronic engineering and the Ph.D. degree in electronic and computer engineering, both from the University of Cagliari, Cagliari, Italy, in 2005 and 2009, respectively. She is a Post doctoral researcher of Automatic Control at the Department of Electrical and Electronic Engineering of the University of Cagliari. She has been a visiting researcher at the University of Illinois (Urbana-Champaign, IL, USA), University of Michigan (Ann Arbor, MI, USA), Universidad de Zaragoza (Spain) and Indiana University Purdue University Indianapolis (Indianapolis, IN, US). She was an instructor at Indiana University Purdue University Indianapolis during the Spring semester 2014. Her research interests are based on discrete event systems, automata, Petri nets, state estimation, diagnosis, identification, supervisory control. She has been the quality manager of the European project FP7-ICT2-3.7 DISC - Distributed Supervisory Control of Large Plants (2008-11). She has been member of the International Program Committee of the 2nd IFAC Conf. on the Analysis and Design of Hybrid Systems (ADHS'06) and of the 18th IEEE Int. Conf. on Emerging Technology and Factory Automation (ETFA2013). She has published 14 international journal papers, 7 book chapters and 35 international conference papers.

Carla Seatzu Carla Seatzu received the Laurea degree in Electrical Engineering and her Ph.D. degree in Electronic and Computer Engineering from the University of Cagliari, Italy, in 1996 and 2000, respectively. Since 2011 she is Associate Professor of Automatic Control at the Department of Electrical and Electronic Engineering of the University of Cagliari, which she joined in 2002 as an Assistant Professor. In 2013 she got the Italian National Abilitation to Full Professor of Automatic Control.

She is Vice-President of the Faculty Committee of Engineering and Architecture and Vice-Coordinator of the Ph.D. Program in Electronic and Computer Engineering at the University of Cagliari.

Carla Seatzu's research interests include discrete-event systems, Petri nets, hybrid systems, networked control systems, manufacturing and transportation systems. She is author of more than 200 publications, including 60+ papers in international journals, 10+ chapters in international books, and one textbook. She is editor of two international books and the proceedings of two international conferences. Her h-index in Scopus is equal to 23.

Actually she is Associate Editor of 4 international journals: IEEE Trans. on Automatic Control, IEEE Trans. on Automation Science, Discrete Event Dynamic Systems, and Nonlinear Analysis: Hybrid Systems. She has also intensively collaborated to the organization of international events. In particular, she is Workshop Chair of the 55th IEEE Conf. on Decision and Control (2015), and was General Co-chair of the 18th IEEE Int. Conf. on Emerging Technologies and Factory Automation (2013).