Diagnosability of event patterns in safe labeled time Petri nets: a model-checking approach

Yannick Pencolé, Audine Subias

▶ **To cite this version:**

Yannick Pencolé, Audine Subias. Diagnosability of event patterns in safe labeled time Petri nets: a model-checking approach. IEEE Transactions on Automation Science and Engineering, 2022, 19 (2), pp.1151 - 1162. ⟨10.1109/TASE.2020.3045565⟩. ⟨hal-03139863⟩

HAL Id: hal-03139863

https://laas.hal.science/hal-03139863v1

Submitted on 12 Feb 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

Diagnosability of event patterns in safe labeled time Petri nets: a model-checking approach

Yannick Pencolé, Audine Subias

***Abstract*—Checking the diagnosability of a time discrete event system usually consists in determining whether a single fault event can always be identified with certainty after a finite amount of time. The aim of this paper is to extend this type of analysis to more complex behaviors, called event patterns, and to propose an effective method to check diagnosability with the use of model-checking techniques. To do so, we propose to convert the pattern diagnosability problem into checking a linear-time property over a specific time Petri net.**

***Note to Practitioners*—This paper is motivated by the problem of improving the monitoring and the supervision of systems like automated and robotised manufacturing systems. Based on a model of the system, the paper proposes a method to assert with certainty whether the available set of sensors will always provide enough information to ensure that a complex and unexpected behavior has not happened in the system. The proposed method uses a publicly available model-checking tool to perform this analysis.**

***Index Terms*—Fault Diagnosis, Diagnosability, Pattern, Time Petri nets, Discrete Event Systems (DES).**

I. Introduction

Diagnosability is the property of a partially observed dynamical system that asserts whether it is always possible to diagnose with certainty the occurrence of an anticipated failure [1], [2]. This paper addresses the problem of checking diagnosability in *timed discrete event systems* such as automated and robotised manufacturing systems, communication protocols. As opposed to most of the previous work the anticipated type of failures are not represented as single fault events but as more complex event assemblings called *event patterns* [3], [4], [5]. This paper proposes a formal and effective method to check pattern diagnosability in timed systems that can be modeled as safe labeled time Petri nets. The first contribution of the paper is a formal characterization of the pattern diagnosis problem in timed system that extends the one presented in [6] for untimed labeled Petri nets. The second contribution is a formal extension of the single fault diagnosability problem as initially defined in [7] to pattern diagnosability. The proposed definition is language-based and does not rely on the actual formalism (time Petri net, time automata,...). We formally prove that this definition encompasses the single fault diagnosability definition of [7]. The third contribution is about a way to turn the pattern diagnosability problem into a model-checking problem [8], [9], [6], by building a specific twin-plant based on a set of products between the system's model and the pattern. The paper proposes an effective solution that uses the off-the-shelf model-checker TINA [10] and presents experimental results.

Yannick Pencolé was with LAAS-CNRS, CNRS, Univ. Toulouse, France, Audine Subias was with LAAS-CNRS, INSA, Univ. Toulouse, France

II. Related work

The problem of fault diagnosis and diagnosability in DESs was introduced a few decades ago in [1], [2] and a survey can be found in [11]. For untimed systems and in the framework of automata, [2] proposed to check the diagnosability by detecting fault-indeterminate cycles in a global diagnoser. The algorithm is however exponential in the number of states of the global behavioral model. Polynomial algorithms have then been proposed, directly based on a synchronisation of two copies of the behavioral model (twin-plant machine in [12] and verifier in [13]), that look for a pair of infinite behaviors of the system with the same observable projection, one faulty and one non-faulty, which proves the system is not diagnosable. More recently, this diagnosability problem has been revisited using Petri nets [14]. It was introduced in [15] for unbounded Petri nets by the use of a diagnoser built from a coverability graph. [16] proposes the definitions of weak and strong diagnosability and solve the problem by using net unfolding techniques. [17] considers a structural approach to give a sufficient condition for diagnosability relying on T-invariants and formulates the problem as a linear programming problem. [18] gives sufficient conditions for diagnosability and undiagnosability of a fault transition based on the notion of the so called g-markings [19], representing a set of inequalities to be solved. [20] updates the verifier to analyse diagnosability on bounded Petri nets based on minimal explanations to reduce the computational complexity. Similarly, minimal explanations are used in [21] for bounded nets and in [22] for unbounded nets, for the construction and the analysis of a compact state space representation called Basis Reachability Graph (BRG) [23]. The approach is extended to labeled Petri nets where different transitions can share the same label in [24]. More recently [25] proposes an approach for bounded and unbounded Petri nets from the coverability graph. [26] proposes some reduction rules to apply on the system model that preserve the diagnosability property and improve the computation efficiency.

For timed discrete event systems (TDES), the literature about fault diagnosis and diagnosability is far less developed. It has been introduced in [7] on timed automata. A fault is said to be diagnosable if there always exists a finite amount of time τ after the occurrence of the fault to assert its occurrence with certainty. If the diagnosability property holds for a fixed time τ, then the system is τ-diagnosable. In [27], another diagnosability property based on time ticks is analyzed: the method relies on a timed product of finite-state generators. In [28], the decidability and computational complexity of the problem is presented. The work of [9] reviews the algorithms for checking diagnosability in automata and timed automata

and shows that both diagnosability problems can be reduced to a Büchi emptyness problem based on a twin-plant as defined in [7]. In the context of Petri nets, the work of [29] deals with P-time Petri nets under partial observation. The method is based on the synthesis of a modified state observer and on a schedulability analysis using linear programming techniques. Finally, [30] proposes a method for the diagnosability analysis of labeled T-time Petri nets under partial observation. The authors develop an approach based on the Modified State Class Graph proposed in [31] to perform τ-diagnosability analysis in two steps: they search for ambiguous sequences and, if such sequences exist, they check for time ambiguities longer than τ. To the best of our knowledge, this last work is the closest to our proposal which consists in extending diagnosability analysis to behavioral event patterns as defined in [6]. The difference is that our proposal deals with the diagnosability problem (looking for a time τ such that the system is τ diagnosable) and not with the τ-diagnosability problem.

The related works presented hereabove only deal with single faults. Diagnosability of more complex behaviors has also been investigated but only for untimed DES. In [32], a pattern is defined as a temporal logic specification over a finite-state automaton. A fault is then defined as a behavior that *does not match* the pattern (that is called a specification in [32]). To check diagnosability, they propose an *ad hoc* algorithm that builds the part of interest of the Kripke structure of the twin plant by synchronizing faulty and non-faulty behaviors. In [3], the patterns are defined as automata and they define the pattern diagnosabililty problem in automata: is it possible to always assert that the system *matched* a pattern after a finite sequence of observations. As in [32] the authors propose an *ad hoc* algorithm that builds the Kripke structure of the twin plant by successive synchronizations. In [33], the same problem is solved in a decentralized manner by aggregating pattern recognizers. In [34], patterns define some contexts represented as regular expressions and the authors propose a method to solve the pattern diagnosis problem on a class of untimed DES called active systems. [4] introduces the notion of patterns into labelled Petri nets. An unfolding approach is then applied on an updated twin-plant. More recently, [6] extends the work of [4] to more sophisticated patterns and proposes an effective method to automatically analyze the diagnosability.

III. Background on Labeled Time Petri Nets

A. Definition and Semantics

The labeled time Petri nets [35] constitute a specific class of Petri nets that maps any transition to a label and a static time interval that rules the transition firing. Let $\mathbb{I}_{\mathbb{Q}^+}$ be the set of nonempty real interval with nonnegative rational end-points.

Definition 1 (LTPN)**.** *A Labeled Time Petri Net (LTPN for short) is a 6-uple $N = \langle P, T, A, E, \ell, I_s \rangle$ such that:*

- *P is a finite set of places;*
- *T is a finite set of transitions ($P \cap T = \varnothing$);*
- *$A \subseteq (P \times T) \cup (T \times P)$ is a binary relation that models the arcs between the places and the transitions;*
- *E is a finite set of transition labels;*
- *$\ell : T \to E$ is a transition labeling function;*
- *$I_s : T \to \mathbb{I}_{\mathbb{Q}^+}$ is a static interval function.*

Throughout this paper, the static interval $I_s(t)$ of a transition t is either closed ($[a, b]$) or semi-closed ($[a, +\infty[$). The *preset* $\mathrm{pre}(t)$ of a transition t is the set of (input) places $\mathrm{pre}(t) = \{p \in P : (p, t) \in A\}$ and the *postset* $\mathrm{post}(t)$ of a transition t is the set of (output) places $\mathrm{post}(t) = \{p \in P : (t, p) \in A\}$. A *state* of an LTPN is a couple $S = \langle M, I \rangle$. M is the marking of the net and is a function (M: $P \to \mathbb{N}$) that assigns to each place a number of tokens (or marks). A marking is usually denoted as a column vector with as many entries as the number of places. All along this paper, we will consider that any LTPN is safe i.e. M: $P \to \{0, 1\}$. I is a partial firing interval application I: $T \to \mathbb{I}_{\mathbb{R}^+}$ that associates to any enabled transition t a time interval of $\mathbb{R}^+$ ($\mathbb{I}_{\mathbb{R}^+}$: intervals with bounds in $\mathbb{R}^+$) in which transition t can be fired. The lower bound of $I(t)$, also called the date of earlier firing, is denoted $\lfloor I(t) \rfloor$, and its upper bound, also called the date of later firing, is denoted $\lceil I(t) \rceil$. A transition t is *enabled* by a given marking M (denoted $t \in enabled(M)$) if and only if $(\forall p \in \mathrm{pre}(t), M(p) = 1)$. An LTPN with an initial marking M_0 is called a marked LTPN. The initial state of a marked LTPN is given by $S_0 = \langle M_0, I_0 \rangle$ where I_0 is defined as follows: for any transition t enabled by M_0, $I_0(t) = I_s(t)$.

Intuitively, in an LTPN a transition is firable from a marking M if it is enabled for a sufficient amount of time i.e an amount of time within its own static time interval. Formally, a net transition t is *firable* from a state $S = \langle M, I \rangle$ at a relative date θ if and only if:

1) $t \in enabled(M)$,
2) $\theta \geq \lfloor I(t) \rfloor$ and for any transition t' enabled by M, θ is not greater than the later firing date of t': $\theta \leq \lceil I(t') \rceil$ if $I(t')$ is right-closed.

Firing a firable transition t at the relative date θ from the state $S = \langle M, I \rangle$ leads to a state $S' = \langle M', I' \rangle$ such that:

- M' is such that $\forall p \in \mathrm{pre}(t) \setminus \mathrm{post}(t), M'(p) = 0$, $\forall p \in \mathrm{post}(t) \setminus \mathrm{pre}(t), M'(p) = 1$, else $M'(p) = M(p)$;
- For any transition $t' \in T$ ($t' \neq t$) enabled by M after the fire of t and enabled by M'
  - if $I(t') = [a, b]$, $I'(t') = [max(0, a - \theta), b - \theta]$;
  - if $I(t') = [a, +\infty[$, $I'(t') = [max(0, a - \theta), +\infty[$.
- else $I'(t') = I_s(t')$.

In the following, the fire of a transition t at the relative date θ is denoted: $\langle M, I \rangle \xrightarrow{\theta t} \langle M', I' \rangle$. A state S is *reachable* in a marked LTPN if it exists a sequence of firable transitions $S_0 \xrightarrow{\theta_1 t_1} S_1 \xrightarrow{\theta_2 t_2} \dots \xrightarrow{\theta_k t_k} S$, also denoted $S_0 \xrightarrow{r} S$ where r is the timed transition sequence $r = \theta_1 t_1 \theta_2 t_2 \dots \theta_k t_k$. The sequence r will be called a *run* of the net. Any run $r' = \theta_1 t_1 \dots \theta_i t_i, i \leq k$ is a *prefix* of r. The set of reachable states from a state S in a marked LTPN N is denoted $R(N, S)$. A marking M' is *reachable* from a marking M (also denoted without ambiguity $M' \in R(N, M)$) if there exists a state $S' = \langle M', I' \rangle$ such that $S' \in R(N, \langle M, I \rangle)$.

B. LTPN extensions and operations

We recall two classical LTPN extensions that will be also used throughout this paper: *inhibitor arc* and *transition pri-*

orities. An inhibitor arc modifies the enabling conditions of transition t that were detailed previously for the so-called regular arcs. With an inhibitor arc (p,t), the input place p must be empty to enable t. Formally, for an LTPN with both types of arcs (regular and inhibitor) the set of input places of a transition t ($\mathrm{pre}(t)$) is then partitioned into $\mathrm{pre}^-(t)$ and $\mathrm{pre}^+(t)$. An arc (p,t) of A is a regular arc if $p \in \mathrm{pre}^+(t)$ and an inhibitor arc if $p \in \mathrm{pre}^-(t)$. A transition t is enabled by a marking M ($t \in enabled(M)$) iff $\forall p \in \mathrm{pre}^-(t), M(p) = 0 \wedge \forall p \in \mathrm{pre}^+(t), M(p) = 1$.

A *priority* defines a binary relation between two transitions of an LTPN, denoted $t_1 \succ t_2$, that is transitive, not reflexive and not symmetrical. A firable transition is not allowed to fire if a transition with a higher priority is firable at the same date.

We finally introduce some further operations on LTPN.

Definition 2 (Restriction of an LTPN N to the transition t)**.** *The* restriction *of the LPTN N to the transition $t \in T$ (inhibitor arcs included) is :*

$$N(t) = \langle \mathrm{pre}(t) \cup \mathrm{post}(t), \{t\}, A_t, \{\ell(t)\}, \ell_t, I_{st} \rangle \quad (1)$$

where $(n_1,n_2) \in A_t$ iff $(n_1,n_2) \in A \wedge (n_1 \in pre(t) \wedge n_2 = t) \vee (n_1 = t \wedge n_2 \in post(t))$.

Definition 3 (Union of LTPNs)**.** *Let $N_i = \langle P_i, T_i, A_i, E_i, \ell_i, I_{si} \rangle, i = \{1,2\}$, the* union *of N_1 and N_2, denoted $N_1 \cup N_2$, is the LTPN*

$$N_1 \cup N_2 = \langle P_1 \cup P_2, T_1 \cup T_2, A_1 \cup A_2, E_1 \cup E_2, \ell, I_s \rangle$$

such that $\forall i \in \{1,2\}$: $\forall t \in T_i, \ell(t) = \ell_i(t)$ and $\forall t \in T_i, I_s(t) = I_{si}(t)$. If N_1 and/or N_2 have inhibitor arcs, there are also in the union $N_1 \cup N_2$, similarly for priorities.

Definition 4 (Difference of LTPNs)**.** *Let $N_i = \langle P_i, T_i, A_i, E_i, \ell_i, I_{si} \rangle, i = \{1,2\}$, the* difference *of N_1 by N_2, denoted $N_1 \setminus N_2$, is the LTPN*

$$N_1 \setminus N_2 = \langle P, T_1 \setminus T_2, A, E_1, \ell_1, I_{s1} \rangle$$

such that:

- $P = P_1 \setminus \{p \in P_2 : \nexists t \in T_1 \setminus T_2, p \in \mathrm{pre}(t) \cup \mathrm{post}(t)\}$;
- $A = A_1 \setminus \{(n_1,n_2) \in A_2 : n_1 \in T_2 \vee n_2 \in T_2\}$;
- $\forall t \in T_1 \setminus T_2, \ell(t) = \ell_1(t),\ I_s(t) = I_{s1}(t)$.

Any inhibitor arc of A_1 in A is retained as an inhibitor arc in A. Any priority in N_1 between two transitions of $T_1 \setminus T_2$ is also kept in $N_1 \setminus N_2$.

C. Timed languages

A *timed sequence* over a set of labels Σ is a sequence ρ of pairs (s,d) where d is a duration and s is a symbol of $\Sigma \cup \{\lambda\}$. Throughout this paper, symbol λ will represent the occurrence of time and will be always considered as a silent event (λ may be part of Σ or not). The set of timed sequences over Σ is denoted $\mathcal{T}(\Sigma)$ and $\mathcal{T}(\Sigma) \subseteq (\Sigma \cup \{\lambda\} \times \mathbb{R}^+)^+$. Each timed sequence ρ has a canonical representation denoted $[\rho]$ with only one λ at the end. Suppose for instance that $\Sigma = \{a,b,c\}$, then $\rho = (a,3).(c,4).(\lambda,2).(b,1).(b,2).(\lambda,4)$ is a timed sequence of $\mathcal{T}(\Sigma)$ also denoted $3a4c2\lambda1b2b4\lambda$. The canonical representation of ρ is $[\rho] = 3a4c3b2b4\lambda$. Throughout this paper, only canonical representations are considered.

Consider now a run r of an LTPN $S_0 \xrightarrow{r} S = \langle M, I \rangle$, we can associate r with a set of canonical timed sequences, denoted $\theta\_\ell(r)$ as follows:

$$\theta\_\ell(r) = \{[\theta\_\ell^{\dashv}(r).d\lambda], d \in [0, d_{\max}[\}, \quad (2)$$

where $\theta\_\ell^{\dashv}(r)$ is the *earliest* timed sequence of r:

- if r is the empty run (no transition) then $\theta\_\ell^{\dashv}(r) = 0\lambda$;
- if $r = \theta_1 t_1 \theta_2 t_2 \ldots \theta_k t_k, k > 0$ then $\theta\_\ell^{\dashv}(r) = [\theta_1 \ell(t_1) \theta_2 \ell(t_2) \ldots \theta_k \ell(t_k) 0\lambda]$.

Date d_{max} is the latest possible duration stay in state S (i.e. the minimal upper bound of the current firing intervals $d_{max} = min_{t \in enabled(M)}(\lceil I(t) \rceil)$). Based on the sets $\theta\_\ell(r)$ we can define a timed language generated by an LTPN. Let Q be a subset of $R(N, M_0)$, a timed sequence belongs to the language generated by the marked LTPN if there is a run of the LTPN that generates the sequence and that leads to a state whose marking is in Q: such a run is called an *admissible run*.

Definition 5 (Language of an LTPN over Q)**.** *The language generated by a marked LTPN N over Q is*

$$\mathcal{L}(N,Q) = \bigcup_{r: S_0 \xrightarrow{r} S = \langle M, I \rangle \wedge M \in Q} \theta\_\ell(r)$$

IV. Problem statement

The fault diagnosis and diagnosability problems have been formally defined with the help of timed automata in [7]. We propose here to extend these definitions to deal with more complex behaviors than single faults.

A. System model

We consider throughout this paper that the supervised system is modelled as a safe LTPN denoted $\Theta = \langle P_\Theta, T_\Theta, A_\Theta, \Sigma_\Theta, \ell_\Theta, I_{\Theta S} \rangle$ with an initial marking denoted $M_{0\Theta}$. Σ_Θ represents the set of event types that are effectively generated by the system ($\lambda \notin \Sigma_\Theta$). In this paper, we assume that according to the instrumentation only a subset of transitions can be observed. These transitions are associated with a sensor that generates an output turned into an observable event by a measurement function. We consider also unobservable transitions that represent effective system's event but not associated with any kind of sensors. Transition labels Σ_Θ are then partitioned into two subsets: Σ_Θ^o is the set of observable events and Σ_Θ^u the set of unobservable ones. Besides this definition, there are several assumptions about the system.

A0 *The system has no deadlock.*

A1 *The system has no zeno runs.*

A2 *From any reachable state, there is no infinite sequence of firable transitions labeled with unobservable events.*

A0 states that for any reachable state S there exists a transition t that is firable from S and such that $\lceil I(t) \rceil \neq \infty$. This classical assumption is from the seminal work [2]. Assumption **A1** states that the model cannot represent unrealistic evolutions of the system by forbiding runs that have an infinite number of events in a finite amount of time [7], [30]. Typically, zeno

runs are modeled by transition loops, each transition being associated with a static time interval whose lower bound is 0. Finally, Assumption **A2** is also a classical assumption for diagnosability analyses [2], [7] without which the system is unlikely diagnosable due to the lack of observations.

Figure 1 presents the model of a system Θ. It consists of two concurrent processes synchronized to perform together a specific activity. The first process (on the left part of the figure) is composed of several activities represented by the places $p_i^1, i \in \{1, \ldots, 5\}$ while the activities of the second process (on the right part of the figure) are modeled by the places $p_i^2, i \in \{1, \ldots, 6\}$. Both processes are synchronized on transition t_1^{12} whose firing leads to the marking of a shared place p^{12} representing the activity that requires the two processes. The end of this activity is modeled by the firing of transition t_2^{12} that leads to the marking of the places p_3^1 and p_6^2 so that each process can evolve independently to perform its own activities. The initial state of each process is given by the initial marking (places with black dots). Transition labels that are in bold represent observable events while the others are unobservable events. It must be noticed that Process 1 (resp. Process 2) only generates $\mathbf{o_1}$ (resp. $\mathbf{o_2}$) events as observations. The timed language generated by the system is

$$\mathcal{L}(\Theta) = \mathcal{L}(\Theta, R(\Theta, M_{0\Theta})) \tag{3}$$

B. Event pattern model

Event patterns gather a set of specific combinations of untimed and unobservable events whose occurrence in the system leads to a situation of interest (i.e. failure, critical/dangerous situation, safe/unsafe behavior,....). Patterns that are studied throughout this paper have been introduced in [6] and are defined as Labeled Petri Nets (LPN), i.e. LTPN such that $I_s(t) = [0, +\infty[$ for any transition t. As for the system model, transitions labels represent events of the system.

Definition 6 (Pattern)**.** *A pattern Ω is an LPN $\Omega = \langle P_\Omega, T_\Omega, A_\Omega, \ell_\Omega, \Sigma_\Omega, M_{0\Omega}\rangle$ associated with a set of final markings $Q_\Omega \subset R(\Omega, M_{0\Omega})$ such that:*

1) *(unobservable) events of Σ_Ω are unobservable ($\Sigma_\Omega \subseteq \Sigma_\Theta^u$);*
2) *(initialized) $M_{0\Omega} \notin Q_\Omega$;*
3) *(well-formed) from any reachable marking M, there exists a run that leads to a marking $M' \in Q_\Omega$;*
4) *(deterministic) from any reachable marking M of $R(\Omega, M_{0\Omega})$ there is no event $e \in \Sigma_\Omega$ that labels more than one enabled transition;*
5) *(stable) from any reachable marking M of $R(\Omega, M_{0\Omega})$ such that $M \in Q_\Omega$ any possible run leads to a marking M' such that $M' \in Q_\Omega$.*

For the sake of simplicity, we denote $\mathcal{L}(\Omega) = \mathcal{L}(\Omega, Q_\Omega)$. Patterns are LPN that aim at representing succinct behaviors that are unobservable (Condition 1) ($\mathcal{L}(\Omega) \subset \Sigma_\Theta^{u*}$). Condition 2 ensures that the pattern does not represent an empty event sequence. Condition 3 guarantees that any run of the pattern is always a prefix of a matching run. Condition 4 avoid ambiguities within the pattern and is required to ensure the correctness of the combination of a pattern and a system as detailed later. Finally Condition 5 ensures that once a pattern has occurred, its effect is definitive (any system's run that has a matching run as a prefix is a matching run).

Figure 2 illustrates different types of patterns of interest for the system of Figure 1. Figure 2a presents k occurrences of a single event b (pattern $\Omega_1^b(k)$). The marking of the final place pp_k^1 indicates the pattern has occurred then the set of final markings is given by $Q_{\Omega_1^b(k)} = \{M\colon M(pp_k^1) = 1\}$. This pattern extends the classical single fault event pattern that is usually studied in the literature. Throughout this paper, we will similarly denote $\Omega_1^f(k)$ the pattern modeling k occurrences of the single event f. Figure 2b presents a pattern that characterizes k occurrences of one event among a predefined set of events, here among $\{b, f\}$. The structure of the pattern depends on the value of k required. If one wants to consider one occurrence of one event among $\{b, f\}$, it is necessary to consider the marking of the place pp_1^1 and then $Q_{\Omega_2(1)} = \{M\colon M(pp_1^2) = 1\}$. The marking of the place pp_2^2 indicates 2 occurrences (2 events b or 2 events f or 1 event b and 1 event f, whatever the ordering) then the set of final markings is $Q_{\Omega_2(2)} = \{M\colon M(pp_2^2) = 1\}$. The number of layers in the pattern depends on the k value. The k occurrences are obtained when the place pp_k^2 is marked hence $Q_{\Omega_2(k)} = \{M\colon M(pp_k^2) = 1\}$. The pattern $\Omega_2(k)$ can be a base to extend the classical notion of fault class with $k = 1$ and the set of events is the set of n fault class events: $\{f_1, f_2, \cdots, f_n\}$. In this case, the pattern has only one layer with as many transitions as fault class events. Finally, Figure 2c presents a pattern that models 3 consecutive occurrences of event b that are not interleaved with any occurrence of event f. This pattern can also be extended into $\Omega_3(k)$ to model k consecutive occurrences of an event without any occurrence of another type of event. This last pattern is interesting as it represents some fairness issues in the system that might be of interest to detect, $Q_{\Omega_3(k)} = \{M\colon M(pp_k^3) = 1\}$.

C. Pattern diagnosis and diagnosability

This subsection introduces the problem of diagnosing patterns in timed discrete event systems and defines the diagnosabilility of a pattern in such a system. For a given pattern Ω, the diagnosis problem consists in defining an Ω-diagnoser function that takes as input a timed sequence of observations and checks for any run of the system that can generate these observations whether it *matches* the pattern Ω or not. It results in the generation of three possible symbols: $\Omega{-}faulty$ (all the possible runs match Ω), $\Omega{-}safe$ (none of them match Ω), $\Omega{-}ambiguous$ (some of them match Ω, some of them do not) [36], [6]. A run r of the system *matches* a pattern Ω if one can find in a word $\rho = \theta_1 e_1 \ldots \theta_n e_n \theta_{n+1} \lambda$ of $\theta\_\ell(r)$ a sub-word $\left(\sum_{i=1}^{j_1} \theta_i\right) e_{j_1} \left(\sum_{i=j_1+1}^{j_2} \theta_i\right) e_{j_2} \ldots \left(\sum_{i=j_{k-1}+1}^{j_k} \theta_i\right) e_{j_k}$ such that $\rho_\Omega = e_{j_1} \ldots e_{j_k}$ is a word of $\mathcal{L}(\Omega)$: this will be denoted by $\rho \Subset \rho_\Omega$ and more generally by $\rho \Subset \Omega$ and $r \Subset \Omega$.

Checking whether a run of Θ can generate the given observations (i.e. the run is consistent with the observations) relies on the timed language projection from the alphabet Σ_Θ to the observable alphabet Σ_Θ^o. Given two alphabets Σ_1, Σ_2,

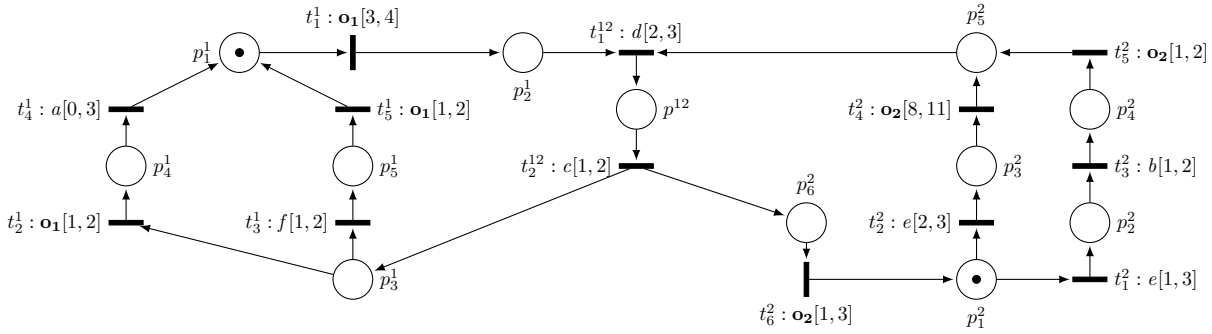


Fig. 1: A system Θ composed of two communicating and partially observable timed processes

the projection $\mathbf{P}_{\Sigma_1 \to \Sigma_2} : \mathcal{T}(\Sigma_1) \to \mathcal{T}(\Sigma_2)$ of a canonical timed sequence is defined as follows.

- $\mathbf{P}_{\Sigma_1 \to \Sigma_2}(\theta\lambda) = \theta\lambda$, for any $\theta \in \mathbb{R}^+$;
- if $e \in \Sigma_2$, $\mathbf{P}_{\Sigma_1 \to \Sigma_2}(\theta e.\rho) = \theta e.\mathbf{P}_{\Sigma_1 \to \Sigma_2}(\rho)$;
- if $e \in \Sigma_1 \setminus \Sigma_2$, two cases hold
  - $\mathbf{P}_{\Sigma_1 \to \Sigma_2}(\theta e\rho) = (\theta + \theta')\lambda$, if $\theta'\lambda = \mathbf{P}_{\Sigma_1 \to \Sigma_2}(\rho)$.
  - $\mathbf{P}_{\Sigma_1 \to \Sigma_2}(\theta e.\rho) = (\theta + \theta')e'.\rho'$, if $\theta' e'.\rho' = \mathbf{P}_{\Sigma_1 \to \Sigma_2}(\rho)$;

The Ω-diagnoser can then be formally defined as follows.

Definition 7 (Ω-diagnoser)**.** *An Ω-diagnoser is a function*

$$\Delta_\Omega : \mathcal{T}(\Sigma_\Theta^o) \to \{\Omega{-}faulty, \Omega{-}safe, \Omega{-}ambiguous\} \text{ with}$$

- $\Delta_\Omega(\sigma) = \Omega{-}faulty$ *if for any* $\rho \in \mathcal{L}(\Theta)$ *that is consistent with* σ *(i.e.* $\mathbf{P}_{\Sigma_\Theta \to \Sigma_\Theta^o}(\rho) = \sigma$*),* $\rho \ni \Omega$*;*
- $\Delta_\Omega(\sigma) = \Omega{-}safe$ *if for any* $\rho \in \mathcal{L}(\Theta)$ *that is consistent with* σ*,* $\rho \notin \Omega$*;*
- $\Delta_\Omega(\sigma) = \Omega{-}ambiguous$ *otherwise.*

It must be noticed that, even if Definition 7 uses Petri nets (Θ and Ω notations), this definition is purely language-based. The diagnosis problem is defined over the set of timed sequences ρ in $\mathcal{L}(\Theta)$, the matching relation is defined over event sequences and consistency is achieved by sequence projection. It follows that Definition 7 defines a problem independently from the underlying model structure but with respect to their language expressiveness (time Petri nets, timed Event Graphs, timed automata...). Considering now a set of patterns $\Omega_1, \ldots, \Omega_n$, the diagnoser function of a system can be defined as Δ: $\mathcal{T}(\Sigma_\Theta^o) \to \prod_{i=1}^n \{\Omega_i{-}faulty, \Omega_i{-}safe, \Omega_i{-}ambiguous\}$ such that $\Delta(\sigma) = (\Delta_{\Omega_1}(\sigma), \ldots, \Delta_{\Omega_n}(\sigma))$.

Let us now illustrate the diagnosis problem with a few examples based on the system of Figure 1 and the family of patterns of Figure 2.

Example 1. *Suppose that the timed sequence that is received by the diagnoser from the system of Figure 1 is* $\sigma_1 = 3\mathbf{o_1}2\mathbf{o_2}4\lambda$ *(that is the reception of event* $\mathbf{o_1}$ *at date 3 then the reception of* $\mathbf{o_2}$ *at date 5 and then nothing till date 9). If the problem is to diagnose the occurrence of one event b, one considers the pattern* $\Omega_1^b(1)$ *with the observation sequence* σ_1*. Then as the event* $\mathbf{o_2}$ *occurs at date 5, transition* t_5^2 *is fired at date 5 which means that event b associated with transition* t_3^2 *has certainly occurred and the result of the Ω-diagnoser is:* $\Delta_{\Omega_1^b(1)}(\sigma_1) = \Omega_1^b(1){-}faulty$*. The pattern* $\Omega_2(1)$ *models one occurrence of an event among* $\{b, f\}$*. In this case at date 9 places* p_5^1 *and* p_6^2 *are marked in the system model, so the event f has certainly occurred (firing of transition* t_3^1*) and* $\Delta_{\Omega_2(1)}(\sigma_1) = \Omega_2(1){-}faulty$*. Finally,* $\Delta_{\Omega_3(2)}(\sigma_1) = \Omega_3(2){-}safe$ *(event b has necessarily occurred once, never twice otherwise a second event* o_2 *would appear in* σ_1 *after date 9 because of* t_6^2*).*

Pattern diagnosability is the property of the system that ensures that, once a pattern Ω has occurred in the system Θ, the Ω-diagnoser is always able to return $\Omega{-}faulty$ after a finite time. Let $time(\rho)$ denote the global duration (i.e. the sum of all durations $time(\rho = \theta_1 e_1 \ldots \theta_n e_n) = \sum_{i=1}^n \theta_i$) of a timed sequence ρ.

Definition 8 (Pattern Diagnosability)**.** *A system Θ is Ω-diagnosable if*

$$\begin{aligned}(\exists \tau \in \mathbb{R}^+), \forall \rho_1, \rho_2 \in \mathcal{L}(\Theta) : \rho_1 = \rho_1' \rho_1'', time(\rho_1'') \geq \tau, \\ \rho_1' \ni \Omega \wedge \mathbf{P}_{\Sigma_\Theta \to \Sigma_\Theta^o}(\rho_1) = \mathbf{P}_{\Sigma_\Theta \to \Sigma_\Theta^o}(\rho_2) \implies \rho_2 \ni \Omega.\end{aligned} \quad (4)$$

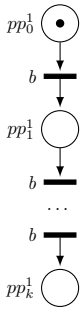
Definition 8 is also language-based and actually extends the single fault diagnosability definition of [7] that is based on time automaton as explained below. Consider a time automaton TA_Θ that generates $\mathcal{L}(\Theta)$. Let $b \in \Sigma_\Theta^u$ be a single fault event, it can be modeled in our framework as the single event pattern represented by $\Omega_1^b(1)$ (see Figure 2a). Then, by definition the following proposition holds.

Proposition 1. *Θ is $\Omega_1^b(1)$-diagnosable iff TA_Θ is diagnosable with respect to the single fault event b in the sense of [7].*

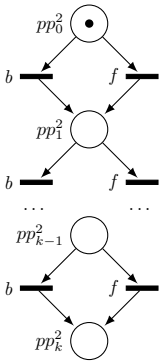
The diagnosability problem consists in finding a finite duration τ after the occurrence of a fault or a pattern in the runs of Θ such that the diagnoser can decide the system is faulty without any ambiguity. In other words, a system is diagnosable iff there exists τ such that the system is τ-diagnosable. [7] introduces such a notion of τ-diagnosability that is also more recently investigated in [30]. We now extend it to pattern diagnosability.

Definition 9 (Pattern τ-Diagnosability)**.** *A system Θ is (Ω, τ)-diagnosable if*

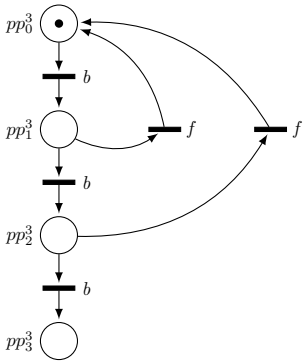
$$\begin{aligned}\forall \rho_1, \rho_2 \in \mathcal{L}(\Theta) : \rho_1 = \rho_1' \rho_1'', time(\rho_1'') \geq \tau, \\ \rho_1' \ni \Omega \wedge \mathbf{P}_{\Sigma_\Theta \to \Sigma_\Theta^o}(\rho_1) = \mathbf{P}_{\Sigma_\Theta \to \Sigma_\Theta^o}(\rho_2) \implies \rho_2 \ni \Omega.\end{aligned}$$



(a) $\Omega_1^b(k)$: k occurrences of b with $Q_{\Omega_1^b(k)} = \{M\colon M(pp_k^1) = 1\}$.



(b) $\Omega_2(k)$: k occurrences of b or f with $Q_{\Omega_2(k)} = \{M\colon M(pp_k^2) = 1\}$.



(c) $\Omega_3(3)$, 3 consecutive occurrences of b without any occurrence of f, $Q_{\Omega_3(3)} = \{M\colon M(pp_3^3) = 1\}$.

Fig. 2: A set of patterns.

Example 2. *Back to the system of Figure 1, the system is $\Omega_1^b(1)$-diagnosable. The occurrence of b (transition t_3^2) can be diagnosed with certainty based on a single observation of the date of the first $\mathbf{o_2}$ (transitions t_4^2 or t_5^2). If the date of the first $\mathbf{o_2}$ is between 3 and 7, event b has definitely occurred (t_5^2 is fired). If no event $\mathbf{o_2}$ is observed between 3 and 7, event b has certainly not happened (t_4^2 is fired). The occurrence of the next $\mathbf{o_2}$ being between 1 and 2 after an occurrence of b (transition t_5^2), it shows that the system is $(\Omega_1^b(1), \tau)$-diagnosable for any $\tau > 2$. In this example, the system is actually $\Omega_1^b(k)$-diagnosable for any k. By construction, Process 2 only generates observable sequences of o_2's and transition t_6^2 is the only transition to be able to generate o_2's at even index in the sequence, so t_6^2 is fully observable, so the date when place p_2^1 gets the token is fully observable and the potential next occurrence of b is diagnosable for the same reasons as above. Similarly as for $k = 1$, the system is $(\Omega_1^b(k), \tau)$-diagnosable for any $\tau > 2$. Consider now the pattern $\Omega_1^f(1)$ where event f replaces event b in Ω_1^b (see Section IV-B). Then, the system is not $\Omega_1^f(1)$-diagnosable. Indeed the occurrence of $o1$ at 2 time units after the marking of p_3^1 can lead either to p_1^1 due to t_5^1 or to t_4^1. Note that if transition $t_4^1 : a[0,3]$ of Figure 1 is restricted to $t_4^1 : a[2,3]$, the system becomes $\Omega_1^f(1)$-diagnosable for the same reasons as b. The system is not $\Omega_2(1)$-diagnosable. As discussed above, event b is diagnosable but it is not sufficient to make $\Omega_2(1)$ diagnosable. The issue is that the occurrence of f may be not diagnosed with certainty while no event b is produced. As long as b does not occur, there always be an ambiguity about the disjunction $\Omega_2(1)$. Note that, with the restriction $t_4^1 : a[2,3]$, the system becomes $\Omega_2(1)$-diagnosable (as there is no more infinite ambiguity about the occurrence of f). Finally, the system is not $\Omega_3(4)$-diagnosable (4 occurrences of event b's without any occurrence of f's) for the same reason as above but with the same restriction to $t_4^1 : a[2,3]$, the system becomes $\Omega_3(4)$-diagnosable.*

Looking back at the system of Figure 1, it can be noticed that each process emits only one type of observable events (process i emits $\mathbf{o_i}$ events). Without observing the effective date of emission of these observations, none of the patterns described above would be diagnosable. Only the observation of time is discriminative here. The remaining question is now how to check that the system Θ is Ω-diagnosable.

V. Pattern diagnosability Analysis

Based on a model Θ and a pattern Ω, this section describes a way to turn the Ω-diagnosability checking problem over Θ into checking a linear-time property over a specific LTPN called a *twin-plant* and denoted Γ. Definition 8 asserts that checking whether a system Θ is Ω-diagnosable consists in determining that there are no infinite runs r and r' in Θ with $r \Vdash \Omega$ and $r' \not\Vdash \Omega$ such that $\mathbf{P}_{\Sigma_\Theta \to \Sigma_\Theta^o}(\theta\_\ell^\dashv(r)) = \mathbf{P}_{\Sigma_\Theta \to \Sigma_\Theta^o}(\theta\_\ell^\dashv(r'))$.

The twin plant gathers into a unique LTPN all the runs r and r' of Θ that generate the same observation sequence. Any run of the twin-plant then characterizes two runs from Θ with the same observations. Finally, checking Ω-diagnosability simply consists in determining in the twin-plant that there is no infinite run that represents a couple of runs (r, r') such that $r \Vdash \Omega$ and $r' \not\Vdash \Omega$. To build the twin-plant, two problems need to be solved.

1) How to represent in an LTPN that any run of Θ matches Ω or not? This is solved by the *pattern matching product* $\Theta \ltimes \Omega$ defined in Section V-A.
2) How to represent in an LTPN two runs of $\Theta \ltimes \Omega$ that generate the same observations? This is solved by applying transition fusions of observable transitions from two copies Π_1 and Π_2 of $\Theta \ltimes \Omega$ (Section V-B).

A. Pattern matching product

The first step of the computation of the twin-plant is to compute an LTPN that represents how a system Θ is actually matching a pattern Ω: this step is the computation of an LTPN called the *pattern matching product* and denoted $\Theta \ltimes \Omega$. This product relies on a specific product between the transitions of Θ and Ω. Let t_Θ be a transition of Θ (with $\ell_\Theta(t_\Theta) = a$, $I_{S\Theta}(t_\Theta) = [\alpha, \beta], \alpha > 0$) [1] with $\mathrm{pre}(t_\Theta) = \{P\Theta_1^1, \ldots, P\Theta_{n\Theta}^1\}$ and $\mathrm{post}(t_\Theta) = \{P\Theta_1^2, \ldots, P\Theta_{m\Theta}^2\}$.

[1] Recall that Assumption **A0** ensures $[\alpha, \beta]$ is a closed interval and **A1** ensures that $\alpha > 0$.

Without loss of generality, we assume that t_Θ is conflict-free (no other transition has the same preset).[2] Let t_Ω be a transition of Ω labeled with a with $\mathrm{pre}(t_\Omega) = \{P\Omega^1_1, \ldots, P\Omega^1_{n\Omega}\}$ and $\mathrm{post}(t_\Omega) = \{P\Omega^2_1, \ldots, P\Omega^2_{m\Omega}\}$. The transition product $t_\Theta \ltimes t_\Omega$ is then the LTPN defined by Figure 3.

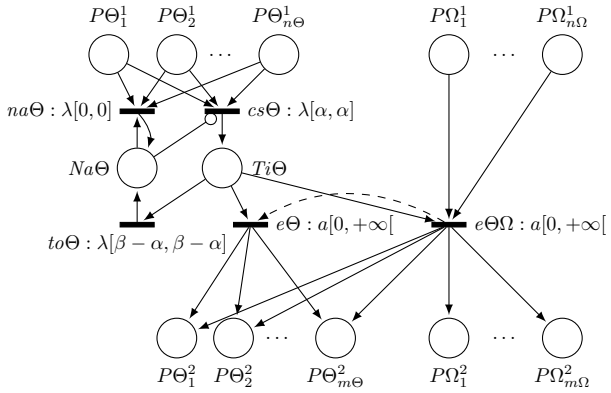


Fig. 3: Pattern matching product between a transition t of Θ ($\ell_\Theta(t) = a$, $I_{s\Theta} = [\alpha, \beta], \alpha > 0$) and a transition of Ω labeled by a.

In time Petri nets compositionnality is kept only when synchronizations involve transitions with an interval $[0, +\infty[$ [37] and transitions with a time interval $[\alpha, \beta]$ cannot be synchronized as is. Indeed, synchronizing two transitions means that it is necessary to determine a clock enabled only when the transitions are both enabled and that stops as soon as one of the local clocks stops. This synchronization is then impossible by simply merging both transitions and assigning a time interval to it based on their respective time intervals as the stop of the synchronized clock depends on the enabling time of each transition and not on the enabling time of their synchronization.

It is then required to perform a *time decomposition* of transition t_Θ to ensure that the synchronization is performed only on transitions with $[0, +\infty[$ time intervals. The time decomposition requires the insertion of transitions with silent events λ (see Section III-C). The result of this time decomposition is the places $Ti\Theta, Na\Theta$ and transitions $na\Theta, cs\Theta, to\Theta, e\Theta$ on the left. This time decomposition ensures that transition $e\Theta$ is firable exactly in the same conditions as the initial transition t_Θ. Indeed, once t_Θ is enabled in Θ by the marking of $P\Theta^1_1, \ldots, P\Theta^1_{n\Theta}$, transition $cs\Theta$ (i.e. *clock start*) is enabled in $\Theta \ltimes \Omega$ and is silenty fired exactly at time $\alpha > 0$ so transition $e\Theta$ starts being firable at time α till time β. At β, either $e\Theta$ or the silent transition $to\Theta$ (i.e. *time out*) is fired, so $e\Theta$ is firable in $[\alpha, \beta]$ as t_Θ and leads to the same postset. Now, as soon as $to\Theta$ is fired, it means that the behavior is actually *not admissible* (transition t_Θ must have been fired in $[\alpha, \beta]$ in Θ as it is conflict-free), the place $Na\Theta$ then permanently holds a token and inhibits any further fire of $cs\Theta$.

[2]Indeed, if t_Θ is in conflict with another transition t'_Θ, Θ can be transformed by adding two silent transitions $\lambda[0, 0]$ enabled by the preset of t_Θ and t'_Θ. Each silent transition has a postset with one new place that replaces the preset of either t_Θ or t'_Θ. New versions of t_Θ and t'_Θ are then conflict-free while the generated langage $\mathcal{L}(\Theta)$ remains the same.

The use of $na\Theta$ is then to empty the preset of $cs\Theta$ in case of a non admissible behavior. The effective synchronization of t_Θ and t_Ω is then represented by transition $e\Theta\Omega$ resulting from the fusion of t_Ω and a copy of the transition $e\Theta$ but as opposed to the classical transition fusion, transition $e\Theta$ is kept to capture the evolutions of the system alone as a pattern evolution corresponds potentially (in case of matching) to a sub-word of the system. A transition priority $e\Theta\Omega \succ e\Theta$ is set between $e\Theta\Omega$ and $e\Theta$ which ensures that if both $e\Theta\Omega$ and $e\Theta$ are firable at a given time, only $e\Theta\Omega$ can be effectively fired. The global definition of the product between a system Θ and a pattern Ω consists then in applying the suitable transition product $t_\Theta \ltimes t_\Omega$ for any couple of transitions (t_Θ, t_Ω) in Θ and Ω that share the same event label. It is formally defined as follows where $\Theta(t_\Theta)$ denotes the restriction of Θ to the transition t_Θ.

Definition 10 (Pattern matching product)**.** *The pattern matching product $\Theta \ltimes \Omega$ is the LTPN such that*

$$\Theta \ltimes \Omega = \Theta \setminus \left(\bigcup_{(t_\Theta, t_\Omega) \in \mathcal{S}} \Theta(t_\Theta) \right) \cup \bigcup_{(t_\Theta, t_\Omega) \in \mathcal{S}} t_\Theta \ltimes t_\Omega$$

where $\mathcal{S} = \{(t_\Theta, t_\Omega) : \ell_\Theta(t_\Theta) = \ell_\Omega(t_\Omega)\}$.

We associate with $\Theta \ltimes \Omega$, the following initial marking $M_{0\Theta\ltimes\Omega}$: for any place p of $\Theta \ltimes \Omega$, $M_{0\Theta\ltimes\Omega}(p) = M_{0\Theta}(p)$ if p is also a place of Θ, $M_{0\Theta\ltimes\Omega}(p) = M_{0\Omega}(p)$ if p is also a place of Ω and $M_{0\Theta\ltimes\Omega}(p) = 0$ otherwise. We denote $Q^{adm}_{\Theta\ltimes\Omega}$ the set of admissible markings, it consists of all the markings M such that for any place of type $Na\Theta$, $M(Na\Theta) = 0$. Let $Q^{match}_{\Theta\ltimes\Omega} \subseteq Q^{adm}_{\Theta\ltimes\Omega}$ denote the set of admissible markings M such that the restriction of the marking M to the places of Ω belongs to the final markings Q_Ω (see Definition 6). It follows that the set of runs in $\Theta \ltimes \Omega$ that lead to a marking of $Q^{match}_{\Theta\ltimes\Omega}$ generates exactly the sub-language of the system that matches the pattern Ω. Formally, we have:

Theorem 1.

1) $\mathcal{L}(\Theta \ltimes \Omega, Q^{adm}_{\Theta\ltimes\Omega}) = \mathcal{L}(\Theta)$;
2) $\mathcal{L}(\Theta \ltimes \Omega, Q^{match}_{\Theta\ltimes\Omega}) = \{\rho \in \mathcal{L}(\Theta) : \rho \Rsh \Omega\}$.

Proof. By construction of the time decomposition (Definition 10), for any transition product $t_\Theta \ltimes t_\Omega$, suppose first that the priority does not exist then $e\Theta$ is firable in $\Theta \ltimes \Omega$ in $[\alpha, \beta]$ from a given date d iff t_Θ is firable in $[\alpha, \beta]$ from d in Θ. By adding the priority, the fire of $e\Theta$ can be possibly replaced by the fire of $e\Theta\Omega$ which generates the same label and only this one as the pattern is deterministic. It follows that $\mathcal{L}(\Theta \ltimes \Omega, Q^{adm}_{\Theta\ltimes\Omega}) = \mathcal{L}(\Theta)$. $Q^{match}_{\Theta\ltimes\Omega}$ gathers the markings of $Q^{adm}_{\Theta\ltimes\Omega}$ whose restrictions to the places of Ω are final markings of Ω. If a run r of $\Theta \ltimes \Omega$ leads to a marking of $Q^{match}_{\Theta\ltimes\Omega}$ it then means it is a run of Θ that generated a word ρ that matches a word ρ_Ω of Ω: $\rho \Rsh \Omega$. Moreover as Ω is stable (Definition 6 condition 5), any continuation of ρ also matches Ω. □

B. Twin plant synthesis

The overall principle is to confront two admissible runs from $\Theta \ltimes \Omega$ that can produce the same observable timed sequence

by duplicating the product $\Theta \ltimes \Omega$. In the following, we denote Π_i, $i \in \{1,2\}$, these two copies. Let $\Theta \ltimes \Omega = \langle P, T, A, \Sigma^u_\Theta \cup \Sigma^o_\Theta \cup \{\lambda\}, \ell, I_s\rangle$ then $\Pi_i = \langle P_i, T_i, A_i, \Sigma_i, \ell_i, I_{si}\rangle$ is defined as follows: $P_i = P \times \{i\}$, $T_i = T \times \{i\}$, $((n,i),(n',i)) \in A_i$ iff $(n,n') \in A$, $\Sigma_i = ((\Sigma^u_\Theta \cup \{\lambda\}) \times \{i\}) \cup \Sigma^o_\Theta$, $\ell_i((t,i)) = \ell(t)$ if $\ell(t) \in \Sigma^o_\Theta$ otherwise $\ell_i((t,i)) = (\ell(t),i)$, $I_{si}((t,i)) = I_s(t)$.

The twin plant is then obtained by applying a transition fusion [38] between a transition t_1 from Π_1 and a transition t_2 from Π_2 that share the same observable label $\ell_1(t_1) = \ell_2(t_2) \in \Sigma^o_\Theta$. For $i = 1,2$, let $\text{pre}(t_i) = \{pi^1_j\}, j \in \{1,\ldots,n_i\}$ and $\text{post}(t_i) = \{pi^2_j\}, j \in \{1,\ldots,m_j\}$. Suppose both transitions are labeled with event $o = \ell_1(t_1) = \ell_2(t_2)$ and their respective static interval is $[\alpha_i, \beta_i], i = 1,2$, the transition fusion denoted $t_1||t_2$ is the LTPN defined in Figure 4.

LTPN $t_1||t_2$ is obtained in two steps. The first intermediate step, similar to the pattern matching product (see Figure 3), is to compute the time decomposition of both transitions t_1 and t_2. Both decompositions independently lead to the creation of two transitions e_j: such a transition e_j, not shown in Figure 4, looks like transition e_Θ in Figure 3, it is labeled with event o and associated with time interval $[0,+\infty[$ and $\text{pre}(e_j) = \{T_{i_j}\}$ and $\text{post}(e_j) = \{pj^2_1,\ldots,pj^2_{m_j}\}$. The second and final step is then to apply the classical transition fusion of e_1 and e_2 to obtain the transition e_{12} of Figure 4.

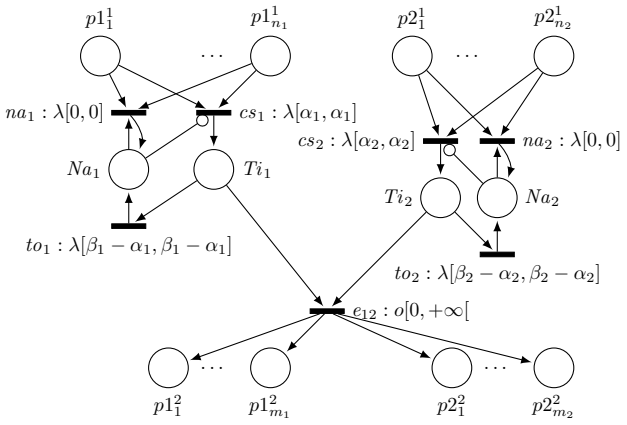


Fig. 4: LTPN $t_1||t_2$ resulting from the transition fusion between a transition t_1 from Π_1 and a transition t_2 from Π_2.

Based on the LTPNs $\Pi_i = \langle P_i, T_i, A_i, \Sigma_i, \ell_i, I_{si}\rangle$, the twin-plant Γ is defined as follows.

Definition 11. *Let $\mathcal{S} = \{(t_1,t_2), t_1 \in T_1, t_2 \in T_2, \ell_1(t_1) = \ell_2(t_2)\}$, the twin plant $\Gamma = (P_\Gamma, T_\Gamma, A_\Gamma, \Sigma_\Gamma, \ell_\Gamma, I_{S\Gamma})$ is:*

$$\Gamma = \bigcup_{i\in\{1,2\}} \left(\Pi_i \setminus \Big(\bigcup_{t:\ell_i(t)\in\Sigma^o_\Theta} \Pi_i(t) \Big) \right) \cup \bigcup_{(t_1,t_2)\in\mathcal{S}} t_1||t_2.$$

Intuitively, the twin plant Γ results from the union of the two copies of the pattern matching products Π_1 and Π_2 where any observable transition t_i is removed and transition fusions are added to Γ: there is one transition fusion per couple of observable transitions (t_1,t_2) from Π_1 and Π_2 respectively sharing the same event label. Let M_i denote a marking of Π_i, then $[\![M_1 \cup M_2]\!]$ denotes the marking function $P_\Gamma \to \mathbb{N}$ such that $\forall p \in P_\Gamma \cap P_{\Pi_1}, [\![M_1 \cup M_2]\!](p) = M_1(p)$ and $\forall p \in P_\Gamma \cap P_{\Pi_2}, [\![M_1 \cup M_2]\!](p) = M_2(p)$. The initial marking of Γ is then $M_{\Gamma 0} = [\![M_{\Pi_1 0} \cup M_{\Pi_2 0}]\!]$ where $M_{\Pi_i 0}$ denotes the initial marking of Π_i and the initial state of Γ is then $S_{\Gamma 0} = \langle M_{\Gamma 0}, I_{S\Gamma 0}\rangle$ with $\forall t \in T_\Gamma, I_{S\Gamma 0}(t) = I_{S\Gamma}(t)$. Finally, Q^{adm}_Γ denotes the set of admissible markings of Γ. Let $Q^{adm}_{\Pi_i}$ denote the set of admissible markings of Π_i, that is the set of markings M_i such that M_i is a restriction of an admissible marking $M \in Q^{adm}_{\Theta\ltimes\Omega}$ with $\forall (p,i) \in P_i, M_i((p,i)) = M(p)$:

$$Q^{adm}_\Gamma = \{[\![M_1 \cup M_2]\!] : M_1 \in Q^{adm}_{\Pi_1} \wedge M_2 \in Q^{adm}_{\Pi_2}\}. \quad (5)$$

Definition 12. *An* admissible run *of Γ is a run from the initial state $S_{\Gamma 0}$ to any state that is composed of an admissible marking from Q^{adm}_Γ.*

Through the rest of this section and to simplify the notations, we introduce the following set of notations. $\mathbf{P}^\Gamma_o$ and $\mathbf{P}^\Theta_o$ respectively stand for their observable projection $\mathbf{P}_{\Sigma_\Gamma \to \Sigma^o_\Theta}$ and $\mathbf{P}_{\Sigma_\Theta \to \Sigma^o_\Theta}$. For $i \in \{1,2\}$, $\mathbf{P}^\Gamma_i$ denotes a special projection of Γ on the LTPN Π_i: let $\rho_i = \mathbf{P}^\Gamma_i(\rho)$ then ρ_i is obtained by computing the classical projection $\rho'_i = \mathbf{P}_{\Sigma_\Gamma \to \Sigma_i}(\rho) \in \mathcal{T}(\Sigma_i)$, and by renaming in ρ'_i any unobservable event $(e,i) \in \Sigma^{uo}_\Theta \times \{i\}$ by $e \in \Sigma^{uo}_\Theta$ which means that $\rho_i \in \mathcal{T}(\Sigma_\Theta)$. We say that a run r of Γ generates a pair (ρ_1,ρ_2) if there exists $\rho \in \theta\_\ell_\Gamma(r)$ such that $\mathbf{P}^\Gamma_1(\rho) = \rho_1$ and $\mathbf{P}^\Gamma_2(\rho) = \rho_2$. The first main property about this twin-plant Γ is that the set of admissible runs of Γ exactly generates the set of pairs (ρ_1,ρ_2) of timed sequences from $\mathcal{L}(\Theta)$ that share the same observable projection, as formally stated in Theorem 2.

Theorem 2.

$$\mathcal{L}(\Gamma, Q^{adm}_\Gamma) = \{\rho \in \mathcal{T}(\Sigma_\Gamma) : \rho_1 = \mathbf{P}^\Gamma_1(\rho) \in \mathcal{L}(\Theta)$$
$$\wedge \rho_2 = \mathbf{P}^\Gamma_2(\rho) \in \mathcal{L}(\Theta) \wedge \mathbf{P}^\Theta_o(\rho_1) = \mathbf{P}^\Theta_o(\rho_2)\}.$$

Proof. See appendix: we need to prove the equality of two languages. Lemma 1 proves the language inclusion $\supseteq$ by the use of Definition 11 and Theorem 1. Lemma 2 proves $\subseteq$ based on Definition 11 and transition fusions. □

For any pair (ρ_1,ρ_2) of timed sequences generated by Γ, as ρ_1 and ρ_2 belong to $\mathcal{L}(\Theta)$, each timed sequence either matches pattern Ω or not.

Definition 13. *A run r of Γ is ambiguous if it is admissible and the pair (ρ_1,ρ_2) with $\rho_i = \mathbf{P}^\Gamma_i(\theta\_\ell^{\dashv}_\Gamma(r)), i \in \{1,2\}$ is such that $\rho_1 \Ni \Omega$ and $\rho_2 \not\Ni \Omega$.*

By Definition 13, we can always associate with an ambiguous run r a pair (ρ_1,ρ_2) such that $\rho_1 \Ni \Omega$ and $\rho_2 \not\Ni \Omega$. But actually, any pair (ρ'_1,ρ'_2) generated by r holds the same property by Proposition 2.

Proposition 2. *If run r is ambiguous then any pair (ρ_1,ρ_2) generated by r is such that $\rho_1 \Ni \Omega$ and $\rho_2 \not\Ni \Omega$.*

Proof. Pattern matching depends on the effective fire of transitions involving events from the pattern and not on time passing only. Timed sequences from $\theta\_\ell_\Gamma(r)$ have the following form: there exists $d \in [0, d_{\max}[$ such that $\rho = [\theta\_\ell^{\dashv}_\Gamma(r) d\lambda]$ (see Section III-C), only the duration d between the last transition fire and the end of the timed sequence is different. □

Definition 14. *An infinite run r_∞ of Γ is an infinite set of finite runs of Γ $\{r_i\}_{i\in\mathbb{N}^+}$ such that for any $i>1$, $S_{\Gamma 0} \xrightarrow{r_i} S_i = S_{\Gamma 0} \xrightarrow{r_{i-1}} S_{i-1} \xrightarrow{\theta_i t_i} S_i$.*

By extension, we say that an infinite run is admissible if any of its finite run is admissible. Finally, an infinite run is ambiguous if there exists $k \in \mathbb{N}^+$ such that $\forall n \geq k$, run r_n is ambiguous. Theorem 3 then holds the fundamental result of the proposed approach.

Theorem 3. *A system Θ is Ω-diagnosable if and only if Γ does not contain ambiguous infinite runs.*

Proof. To prove the theorem, we are going to actually prove that a system Θ is not Ω-diagnosable if and only if Γ contains an ambiguous infinite run.

($\Rightarrow$) First, let us remark that, Equation (4) from Definition 8 is equivalent to

$$\begin{aligned}\forall \rho_1,\rho_2 \in \mathcal{L}(\Theta), \exists \tau \in \mathbb{R}^+ : \rho_1 = \rho_1'\rho_1'', time(\rho_1'') \geq \tau,\\ \rho_1' \ni \Omega \wedge \mathbf{P}_o^\Theta(\rho_1) = \mathbf{P}_o^\Theta(\rho_2) \implies \rho_2 \ni \Omega.\end{aligned} \tag{6}$$

The fact that (4) $\Rightarrow$ (6) is straightforward. Then, from Equation (6), Equation (4) is obtained by simply selecting the maximal value τ amongst the possible values of τ's (therefore (4) $\Leftarrow$ (6)). Assuming Θ is not Ω-diagnosable, then from (6):

$$\begin{aligned}\exists \rho_1,\rho_2 \in \mathcal{L}(\Theta), \forall \tau \in \mathbb{R}^+ : \rho_1 = \rho_1'\rho_1'', time(\rho_1'') \geq \tau,\\ \rho_1' \ni \Omega \wedge \mathbf{P}_o^\Theta(\rho_1) = \mathbf{P}_o^\Theta(\rho_2) \wedge \rho_2 \not\ni \Omega.\end{aligned} \tag{7}$$

Let $\rho_1,\rho_2 \in \mathcal{L}(\Theta)$ be such that $\rho_1 = \rho_1'\rho_1''$, $\rho_1' \ni \Omega$, $\mathbf{P}_o^\Theta(\rho_1) = \mathbf{P}_o^\Theta(\rho_2) \wedge \rho_2 \not\ni \Omega$ (such a pair exists as Θ is not Ω-diagnosable). Any prefix σ_1 of ρ_1 that keeps ρ_1' as a prefix is such that $\sigma_1 \ni \Omega$. Any prefix σ_2 of ρ_2 is such that $\sigma_2 \not\ni \Omega$. Consider now any pair (σ_1,σ_2) such that $time(\sigma_1) = time(\sigma_2)$, as $\mathbf{P}_o^\Theta(\rho_1) = \mathbf{P}_o^\Theta(\rho_2)$, then $\mathbf{P}_o^\Theta(\sigma_1) = \mathbf{P}_o^\Theta(\sigma_2)$. By Theorem 2 and Proposition 2, there must exist an admissible and ambiguous run r of Γ that generates (σ_1,σ_2). As Θ is not Ω-diagnosable, ρ_1 and ρ_2 can be arbitrarily long (i.e $\forall \tau \in \mathbb{R}^+$ there exist ρ_1 and ρ_2 such that $time(\rho_1) \geq \tau$ and $time(\rho_2) \geq \tau$). Assumption **A0** ensures that the number of events in ρ_1 and ρ_2 is not bounded. Moreover Assumption **A2** ensures that the number of observable events in ρ_1 and ρ_2 is not bounded. It follows that it must exist at least an infinite run r_∞ in Γ that is ambiguous: r_∞ is defined as the infinite set of ambiguous runs r that generate the set of prefixes (σ_1,σ_2).

($\Leftarrow$) Suppose that Γ has an ambiguous infinite run r_∞. By Assumption **A1**, $\lim_{i\to+\infty} time(\theta\_\ell_\Gamma^{\dashv}(r_i)) = +\infty$ and by assumption **A2** $\lim_{i\to+\infty} |\mathbf{P}_{\Sigma_\Gamma\to\Sigma_\Gamma^o}(\theta\_\ell_\Gamma^{\dashv}(r_i))| = +\infty$. Let $k \in \mathbb{N}^+$ be the lowest index such that any run $r_n \in r_\infty, n \geq k$ is ambiguous. By Definition 13, there exists an ambiguous pair (ρ_1^k,ρ_2^k) generated from $\theta\_\ell_\Gamma^{\dashv}(r_k)$. Let us suppose without loss of generality that $\rho_1^k \ni \Omega$ and $\rho_2^k \not\ni \Omega$. By Proposition 2, any $(\rho_1^k\theta^k\lambda, \rho_2^k\theta^k\lambda)$ generated from $\theta\_\ell_\Gamma(r_k)$ is also ambiguous and $\rho_1^k\theta^k\lambda \ni \Omega$, $\rho_2^k\theta^k\lambda \not\ni \Omega$. r_{k+1} is also ambiguous, so there also exists the ambiguous pair $(\rho_1^{k+1},\rho_2^{k+1})$ generated from $\theta\_\ell_\Gamma^{\dashv}(r_{k+1})$. If $\ell_\Gamma(t_{k+1}) \neq \lambda$, $\rho_1^{k+1} = \rho_1^k\theta^{k+1}\mathbf{P}_1^\Gamma(\ell_\Gamma(t_{k+1}))0\lambda$, otherwise $\rho_1^{k+1} = \rho_1^k\theta^{k+1}\lambda$. Similarly, if $\ell_\Gamma(t_{k+1}) \neq \lambda$, $\rho_2^{k+1} = \rho_2^k\theta^{k+1}\mathbf{P}_2^\Gamma(\ell_\Gamma(t_{k+1})0\lambda$, otherwise $\rho_2^{k+1} = \rho_2^k\theta^{k+1}\lambda$. By Proposition 2, any $(\rho_1^{k+1}\theta^{k+1}\lambda, \rho_2^{k+1}\theta^{k+1}\lambda)$ generated from $\theta\_\ell_\Gamma(r_{k+1})$ is also ambiguous and $\rho_1^{k+1}\theta^{k+1}\lambda \ni \Omega$, $\rho^{k+1}\theta_2^{k+1}\lambda \not\ni \Omega$. The infinite set of pairs $\{(\rho_1^i,\rho_2^i)\}_{i\geq k}$ can be inductively defined as above. It follows that $\forall t \in \mathbb{R}^+$, there exists $(\rho_1^i\theta^i\lambda, \rho_2^i\theta^i\lambda)$ such that $\rho_1^i\theta^i\lambda = \rho_1^k\rho_1'$ with $\rho_1^k \ni \Omega$, $t = time(\rho_1^i\theta^i\lambda) - time(\rho_1^k)$ and $\rho_1^i\theta^i\lambda \ni \Omega$ but $\rho_2^i\theta^i\lambda \not\ni \Omega$. Therefore, the system Θ is not Ω-diagnosable. □

VI. A MODEL-CHECKING PROCEDURE FOR CHECKING PATTERN DIAGNOSABILITY IN LTPN

A. Model-checking problem

Checking Ω-diagnosability of the system Θ is equivalent to checking that there is no infinite run that is ambiguous in Γ. We propose to solve this problem by using the LTPN model checker TINA [10], [39]. TINA is able to edit LTPN and verify SE-LTL properties (State/Event Linear Temporal Logic) over LTPN. A formula ψ is a SE-LTL formula if it is a universally quantified formula $\psi ::= \forall\varphi$ where $\varphi ::= cst \mid r \mid \neg\varphi \mid \varphi\vee\varphi \mid \varphi\wedge\varphi \mid \bigcirc\varphi \mid \Box\varphi \mid \Diamond\varphi \mid \varphi\mathbf{U}\varphi$. The constant cst can be $\bot$ (false), $\top$ (true), $\mathbf{dead}$ (deadlock), $\mathbf{div}$ (temporal divergence), $\mathbf{sub}$ (partially known state). r defines constraints $e \bigtriangleup e$ with $\bigtriangleup \in \{=,<,>,\leq,\geq\}$ between arithmetical expressions e involving place and transition symbols from the underlying LTPN. The operators $\bigcirc$ (next), $\Box$ (always), $\Diamond$ (eventually) and $\mathbf{U}$ (until) have their usual LTL semantics. Based on SE-LTL, it is in particular possible to implicitly characterize sets of markings in a given LTPN. For instance, suppose an LTPN N composed of $n \geq 2$ places $p_1,p_2,\ldots,p_n$. The set Q of markings $M \in Q$ of N such that $M(p_1)=1$ and $M(p_2)=1$ can be characterized by the SE-LTL property denoted $\textsc{Markings}(N,Q)$:

$$\textsc{Markings}(N,Q) \equiv (p_1 = 1 \wedge p_2 = 1). \tag{8}$$

To implement the resolution of the problem as a model-checking problem, it then suffices to design the SE-LTL formula $\textsc{Diagnosable}(\Theta,\Omega)$ that asserts there is no ambiguous infinite run in Γ. This formula is detailed below step by step. We recall here that the twin plant Γ results from the synchronization of two copies $\Pi_i = \Theta_i \ltimes \Omega_i, i \in \{1,2\}$.

Diagnosability involves runs in Γ only composed of admissible markings M. A marking M_i from Π_i is not admissible as soon as for one place of type $Na\Theta$ (see Section V-A), we have $M_i(Na\Theta) \neq 0$. Then the following SE-LTL properties $\textsc{Adm}(\Pi_i)$ (resp. $\textsc{Adm}(\Gamma)$) characterize the set of admissible markings of Π_i (resp. Γ):

$$\textsc{Adm}(\Pi_i) \equiv \bigwedge_{Na\Theta\in\Pi_i} Na\Theta = 0. \tag{9}$$

$$\textsc{Adm}(\Gamma) \equiv \bigwedge_{i=\{1,2\}} \textsc{Adm}(\Pi_i). \tag{10}$$

Now, $\textsc{Match}(\Pi_i)$ is the property of any marking M of Π_i resulting from a pattern matching, firstly M is admissible and secondly its restriction to the places of Ω_i should hold the properties of the admissible marking set Q_{Ω_i}:

$$\textsc{Match}(\Pi_i) \equiv \textsc{Adm}(\Pi_i) \wedge \textsc{Markings}(\Omega_i, Q_{\Omega_i}). \tag{11}$$

Similarly, the property $\text{NoMatch}(\Pi_i)$ asserts the admissible marking M of Π_i is not a matching property, i.e. its restriction $M_{|\Omega_i}$ does not belong to the final markings of Ω_i:

$$\text{NoMatch}(\Pi_i) \equiv \text{Adm}(\Pi_i) \wedge \neg \text{Markings}(\Omega_i, Q_{\Omega_i}). \quad (12)$$

Now, a marking M of Γ is said to be ambiguous, if its restriction M_1 to Π_1 is a matching marking but its restriction M_2 to Π_2 is not a matching property:

$$\text{Ambiguous}(\Gamma) \equiv \text{Match}(\Pi_1) \wedge \text{NoMatch}(\Pi_2). \quad (13)$$

To finally setup $\text{Diagnosable}(\Theta, \Omega)$, the idea is to write that any ambiguous run is finite. So we first need to logically characterize a finite ambiguous run. As soon as the run contains an ambiguous marking M, two cases hold. Either it will eventually reach ($\Diamond$) an admissible marking M' that is not ambiguous which means that the restriction M'_2 of M' is a matching marking for Π_2 (as it is already for Π_1):

$$\Box\text{Ambiguous}(\Gamma) \Rightarrow \Diamond(\text{Adm}(\Gamma) \Rightarrow \text{Match}(\Pi_2)); \quad (14)$$

or it will eventually reach a dead-end

$$\Box\text{Ambiguous}(\Gamma) \Rightarrow \Diamond(\mathbf{dead}). \quad (15)$$

Theorem 3 can then be translated based on expressions (14) and (15) into the following SE-LTL property on Γ.

Corollary 1. *The system Θ is Ω-diagnosable iff the universally quantified formula* $\text{Diagnosable}(\Theta, \Omega)$ *is true:*

$$\begin{aligned}\text{Diagnosable}(\Theta, \Omega) \equiv \forall\Box\text{Ambiguous}(\Gamma) \Rightarrow \\ \Diamond(\text{Adm}(\Gamma) \Rightarrow (\text{Match}(\Pi_2) \vee \mathbf{dead})).\end{aligned} \quad (16)$$

B. Algorithm and complexity analysis

Algorithm 1 summarizes the method. Line 1 implements the different steps to compute Γ from Θ and Ω as detailed in Section V. Line 2 computes the query based on Γ that is presented in Corollary 1. Line 3 calls the model-checker on Γ with the query. The model-checker returns as a response either **true** or **false** and in the latter case a counter-example that is an infinite admissible *run* of Γ that is ambiguous (see Definitions 13 and 14). Line 5 retrieves from this run the involved couple of infinite runs (r_1, r_2) from Θ.

Algorithm 1: Ω-diagnosability analysis

input : A system Θ and a pattern Ω

1 $\Gamma \leftarrow$ `TwinPlantSynthesis(`Θ, Ω`)`;

2 $\text{Diagnosable}(\Theta, \Omega) \leftarrow$ `QuerySynthesis(`Γ`)`;

3 $(resp, run) \leftarrow$ `CallTina(`$\Gamma, \text{Diagnosable}(\Theta, \Omega)$`)`;

4 **if** $resp = \mathbf{true}$ **then return true**;

5 **else return** $(r_1, r_2) \leftarrow$ `RunExtractions(`run`)`;

output: true iff Θ is Ω-diagnosable

output: If not Ω-diagnosable, a pair of infinite non-diagnosable runs (r_1, r_2) of Θ.

The overall complexity of Algorithm 1 is the complexity of the model-checking part of the algorithm that is performed at line 3. Line 3 actually checks the property $\text{Diagnosable}(\Theta, \Omega)$ on the Strong State Class Graph of Γ (SSCG). The number of states of the SSCG is in $O(2^{|P_\Gamma|})$. By construction of Γ (Line 1), $|P_\Gamma|$ linearly depends on $|P_\Theta|$ and $|P_\Omega|$ which means that the SSCG is in $O(2^{|P_\Theta|})$ and in $O(2^{|P_\Omega|})$. As stated in [40], checking the satisfiability of a formula φ in Γ is in $O(2^{|P_\Gamma|} \times 2^{|\varphi|}) = O(2^{|P_\Theta|} \times 2^{|\varphi|})$ where $|\varphi|$ is the length of the formula φ. In our case, $|\varphi| = |\text{Diagnosable}(\Theta, \Omega)|$ is in $O(|Q^{adm}_\Gamma|) = O(2^{|P_\Theta|})$ and in $O(|Q^{adm}_\Omega|) = O(2^{|P_\Omega|})$. To summarize, Algorithm 1 is in the worst case in $O(2^{2^{|P_\Theta|}})$ and in $O(2^{2^{|P_\Omega|}})$.

C. Experimental results

We report on the experimental results of the 8 diagnosability analyses described in Example 2. Table I recalls the 8 scenarios. Two scenarios are based on the system Θ where transition $a[0,3]$ has been replaced by $a[2,3]$. As the complexity of the algorithm depends on the number of places in the nets and on the size of the query (16) of Corollary 1, we first apply a few optimizations in the proposed algorithm. Firstly, we optimize the number of non-admissible places $Na\Theta$ in the twin. In theory, there are as many places $Na\Theta$ as there are time decompositions in the system, however as $Na\Theta$ is used to assert whether the current marking is admissible or not, it is actually possible to merge all the places $Na\Theta$ into one and only one so in the resulting twin, there are only two places namely $Na\Theta_1$ and $Na\Theta_2$ to represent the non-admissible runs of the duplicated systems Π_1 and Π_2 (see Section V-B). Secondly, to reduce the size of the query (16), we do not encode $\text{Ambiguous}(\Gamma)$ (eq (13)) but only $\text{Match}(\Pi_1) \wedge \text{Adm}(\Pi_2)$ as it is sufficient: we let the model checker handle the case where $\text{NoMatch}(\Pi_2)$ is false (i.e. no ambiguity so no diagnosability issue) in the premise which implies the global question is true (implication). We also do not encode $\text{Adm}(\Pi_2)$ required in $\text{Match}(\Pi_2)$ (eq (11)) in the conclusion of the implication as it is implicit (already in the premise of the implication in $\text{Adm}(\Gamma)$). Table II presents the experimental results for these scenarios with the optimized algorithm. For instance, consider Case 5 where the pattern is $\Omega_2(1)$ with $Q_{\Omega_2(1)} = \{M{:}M(pp^2_1) = 1\}$ (see Figure 2b, $k = 1$), the actual optimized query used to get the results in

Case	Pattern	System
1	$\Omega^b_1(1)$: one event b	Θ
2	$\Omega^b_1(2)$: two events b	Θ
3	$\Omega^b_1(10)$: ten events b	Θ
4	$\Omega^f_1(1)$: one event f	Θ
5	$\Omega_2(1)$: one event f or one event b	Θ
6	$\Omega_2(1)$: one event f or one event b	Θ: $a[0,3] \rightarrow a[2,3]$
7	$\Omega_3(4)$: 4 consecutive events b without an f	Θ: $a[0,3] \rightarrow a[2,3]$
8	$\Omega_3(4)$: 4 consecutive events b without an f	Θ

TABLE I: Diagnosability analyses described in Example 2.

Table II is:

$$\text{DIAGNOSABLE}(\Theta, \Omega_2(1)) \equiv$$
$$\Box(Na\Theta_1 = 0 \wedge Na\Theta_2 = 0 \wedge pp1_1^2 = 1) \Rightarrow$$
$$(\Diamond((Na\Theta_1 = 0 \wedge Na\Theta_2 = 0) \Rightarrow (pp2_1^2 = 1 \vee \mathbf{dead}))) \quad (17)$$

Place name ppi_l^2 denotes the place pp_l^2 in the corresponding product Π_i (they are pattern places inside Π_i). $\text{MATCH}(\Pi_1) \wedge \text{ADM}(\Pi_2)$ is encoded with $(Na\Theta_1 = 0 \wedge pp1_1^2 = 1 \wedge Na\Theta_2 = 0)$. $pp2_1^2 = 1$ is the optimized way to encode $\text{MATCH}(\Pi_2)$. Table II presents for each scenario the size of the twin Γ, the size of the strong state class graph computed by TINA during the verification and the overall time for the computation of the results (time that measures the complete computational time of Algorithm 1). A scenario labeled with OK is a diagnosable scenario, otherwise it is labeled with KO. All the results are consistent with the intuition described in Example 2, the most complex one being obtained in less than 25s in one core of a Ryzen 7 1770 2.2Ghz. We finally recall that, without any quantitative information about time, none of these scenarios would be diagnosable. This explains the size of the SSGs which implicitly represent all the time interleavings of events.

	Γ Size (pl/tr/arcs)	SSG st.	SSG tr.	Time (s)	Result
1	44/76/224	65101	188706	1.931	OK
2	46/78/232	101671	290471	2.918	OK
3	62/94/296	394231	1104591	12.074	OK
4	44/76/224	85107	231247	1.957	KO
5	46/84/248	83143	246688	2.560	KO
6	46/84/248	51734	146106	1.518	OK
7	52/94/288	145627	390583	4.163	OK
8	52/94/288	705376	2012231	21.506	KO

TABLE II: Experimental results for Example 2.

VII. CONCLUSION

This paper introduces pattern diagnosability for timed discrete event systems as an extension of the single fault diagnosability problem. A fully automated approach to check the problem over safe LTPN is then formally detailed and fully implemented as a model-checking problem. One of the main difficulties is to formally characterize the occurrence of a pattern which is achieved by the definition of an original pattern matching product. In our perspectives, we plan to extend this approach to also deal with time patterns and characterize specific subclasses of decidable problems in LTPN. We also plan to update the model-checker to specifically deal with twin-plant construction as initiated in our recent work [41].

APPENDIX

Lemma 1. *For any timed sequences $\rho_1, \rho_2 \in \mathcal{L}(\Theta)$ such that $\mathbf{P}_o^\Theta(\rho_1) = \mathbf{P}_o^\Theta(\rho_2) = \sigma$ there exists a timed sequence $\rho \in \mathcal{L}(\Gamma, Q_\Gamma^{adm})$ such that $\mathbf{P}_o^\Gamma(\rho) = \sigma$, $\mathbf{P}_i^\Gamma(\rho) = \rho_i, i \in \{1, 2\}$.*

Proof. Theorem 1 states that one can find a run r_i in Π_i such that $\rho_i' \in \theta\_\ell_{\Pi_i}(r_i)$ and ρ_i is just as ρ_i' by renaming the unobservable event (e, i) from ρ_i' to e. As $\mathbf{P}_o^\Theta(\rho_1) = \mathbf{P}_o^\Theta(\rho_2) = \sigma$, it follows that both runs r_i fire observable transitions not only at the same date but also with the same label. By definition of Γ (Definition 11), the firing conditions of any unobservable transitions from both Π_i remain the same in Γ. Consider t_1 from r_1 and t_2 from r_2 be the first couple of observable transitions that is fired. The transition fusion in Γ ensures that the transition e_{12} of $t_1||t_2$ (see Figure 4) can be fired at the same date and its postset marking remains admissible. Therefore, this postset marking gathers the postset markings of t_1 and t_2 which means that the firing conditions of next transitions in r_1 and r_2 remain the same in Γ. Applying inductively the same reasoning to the sequence of couples (t_1, t_2) of observable transitions fired at the same time in r_1 and r_2, it follows that we can design an admissible run r of Γ such that $\mathbf{P}_o^\Gamma(\rho) = \sigma$, $\mathbf{P}_i^\Gamma(\rho) = \rho_i, i \in \{1, 2\}$. □

Lemma 2. *For any timed sequence $\rho \in \mathcal{L}(\Gamma, Q_\Gamma^{adm})$ there exists a couple of timed sequences $\rho_1, \rho_2 \in \mathcal{L}(\Theta)$ such that $\mathbf{P}_i^\Gamma(\rho) = \rho_i, i \in \{1, 2\}$; and $\mathbf{P}_o^\Gamma(\rho) = \mathbf{P}_o^\Theta(\rho_1) = \mathbf{P}_o^\Theta(\rho_2)$.*

Proof. As $\rho \in \mathcal{L}(\Gamma, Q_\Gamma^{adm})$, there exists an admissible run $S_{\Gamma 0} \xrightarrow{r} S$ in Γ such that $\rho \in \theta\_\ell_\Gamma(r)$. By Definition 11, r is a sequence composed of transitions either from the products $\Pi_{i\in\{1,2\}}$ or being transition fusions like $t_1||t_2$ where t_i is in Π_i. Consider now $t_1||t_2$ as the first transition fusion present in r. As r is admissible, there must be only one fire of $cs_{i\in\{1,2\}}$ followed by one fire of e_{12} (transitions from $t_1||t_2$, (see Figure 4)) associated with this first transition fusion $t_1||t_2$. By definition of Γ (Definition 11), the firing condition of any unobservable transition from $\Pi_{i\in\{1,2\}}$ remain the same in Γ, so the sequence r_i^0 of any unobservable transition fired in r from Π_i before the fire of transitions $cs\Theta_i$ can be fired in the same conditions in Π_i. As the preset of $cs\Theta_i$ is the preset of t_i, it means that t_i is enabled in Π_i at the same time than $cs\Theta_i$ in Γ. As e_{12} is fired in r it means that t_i is firable at the same absolute date than e_{12} after the run r_i^0 in Π_i. The postset marking of e_{12} being admissible, it also gathers the postset markings of t_i from Π_i: it follows that we can design a run r_i^0 followed by the fire of t_i in Π_i that leads to the marking M_i included in the marking of Γ after the fire of e_{12} in r. Applying inductively this reasoning on the sequence of transition fusions present in r, it follows that we can design a couple of runs r_1 and r_2 from r such that there exists $\rho_i' \in \theta\_\ell_{\Pi_i}(r_i)$ and $\mathbf{P}_o^{\Pi_i}(\rho_i') = \mathbf{P}_o^\Gamma(\rho)$. The result then follows from Theorem 1. □

REFERENCES

[1] F. Lin, "Diagnosability of discrete event systems and its applications," *Discrete Event Dynamic Systems*, vol. 4, no. 2, pp. 197–212, May 1994.

[2] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1555–1575, Sep. 1995.

[3] T. Jéron, H. Marchand, S. Pinchinat, and M.-O. Cordier, "Supervision patterns in discrete event systems diagnosis," in *8th International Workshop on Discrete Event Systems*, Jul. 2006, pp. 262–268.

[4] H.-E. Gougam, A. Subias, and Y. Pencolé, "Supervision Patterns: Formal Diagnosability Checking by Petri Net Unfolding," *IFAC Proceedings Volumes*, vol. 46, no. 22, pp. 73–78, Jan. 2013.

[5] A. Boussif, M. Ghazel, and J. C. Basilio, "Intermittent fault diagnosability of discrete event systems: an overview of automaton-based approaches," *Discrete Event Dynamic Systems*, 2020.

[6] H.-E. Gougam, Y. Pencolé, and A. Subias, "Diagnosability analysis of patterns on bounded labeled prioritized Petri nets," *Discrete Event Dynamic Systems*, vol. 27, no. 1, pp. 143–180, Mar. 2017.

[7] S. Tripakis, "Fault Diagnosis for Timed Automata," in *Formal Techniques in Real-Time and Fault-Tolerant Systems*, ser. Lecture Notes in Computer Science, W. Damm and E. R. Olderog, Eds. Springer Berlin Heidelberg, 2002, pp. 205–221.

[8] C. Pecheur and A. Cimatti, "Formal Verification of Diagnosability via Symbolic Model Checking," in *Proceedings of the 18th International Joint Conferences on Artificial Intelligence*, Acapulco, Mexico, 2003, pp. 363–369.

[9] F. Cassez, "A note on fault diagnosis algorithms," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, Dec. 2009, pp. 6941–6946.

[10] B. Berthomieu, P.-O. Ribet, and F. Vernadat, "The tool TINA – Construction of abstract state spaces for petri nets and time petri nets," *International Journal of Production Research*, vol. 42, no. 14, pp. 2741–2756, Jul. 2004.

[11] J. Zaytoon and S. Lafortune, "Overview of fault diagnosis methods for Discrete Event Systems," *Annual Reviews in Control*, vol. 37, no. 2, pp. 308–320, Dec. 2013.

[12] S. Jiang, Z. Huang, V. Chandra, and R. Kumar, "A polynomial algorithm for testing diagnosability of discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 46, no. 8, pp. 1318–1321, Aug. 2001.

[13] T.-S. Yoo and S. Lafortune, "Polynomial-time verification of diagnosability of partially observed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 47, no. 9, pp. 1491–1495, Sep. 2002.

[14] F. Basile, "Overview of fault diagnosis methods based on Petri net models," in *European Control Conference (ECC)*, Jun. 2014, pp. 2636–2642.

[15] T. Ushio, I. Onishi, and K. Okuda, "Fault detection based on Petri net models with faulty behaviors," in *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 1, Oct. 1998, pp. 113–118 vol.1.

[16] S. Haar, A. Benveniste, E. Fabre, and C. Jard, "Partial order diagnosability of discrete event systems using petri net unfoldings," in *42nd IEEE International Conference on Decision and Control*, vol. 4, Dec. 2003, pp. 3748–3753 vol.4.

[17] YuanLin Wen, ChunHsi Li, and MuDer Jeng, "A polynomial algorithm for checking diagnosability of Petri nets," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, Oct. 2005, pp. 2542–2547 Vol. 3.

[18] F. Basile, P. Chiacchio, and G. De Tommasi, "Online diagnosis of discrete event systems based on Petri nets," in *2008 9th International Workshop on Discrete Event Systems*, May 2008, pp. 436–442.

[19] ——, "An Efficient Approach for Online Diagnosis of Discrete Event Systems," *IEEE Transactions on Automatic Control*, vol. 54, no. 4, pp. 748–759, Apr. 2009.

[20] G. Jiroveanu and R. K. Boel, "The Diagnosability of Petri Net Models Using Minimal Explanations," *IEEE Transactions on Automatic Control*, vol. 55, no. 7, pp. 1663–1668, Jul. 2010.

[21] M. P. Cabasino, A. Giua, and C. Seatzu, "Diagnosis of discrete event systems using labeled Petri nets," *IFAC Proceedings Volumes*, vol. 42, no. 5, pp. 52–57, Jun. 2009.

[22] M. P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu, "Diagnosability analysis of unbounded Petri nets," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 28th Chinese Control Conference*, Dec. 2009, pp. 1267–1272.

[23] M. P. Cabasino, A. Giua, and C. Seatzu, "Fault detection for discrete event systems using Petri nets with unobservable transitions," *Automatica*, vol. 46, no. 9, pp. 1531–1539, Sep. 2010.

[24] ——, "Diagnosability of Discrete-Event Systems Using Labeled Petri Nets," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 1, pp. 144–153, Jan. 2014.

[25] D. Lefebvre and E. Leclercq, "Diagnosability of Petri nets with observation graphs," *Discrete Event Dynamic Systems*, vol. 26, no. 3, pp. 539–559, Sep. 2016.

[26] B. Li, M. Khlif-Bouassida, and A. Toguyéni, "Reduction rules for diagnosability analysis of complex systems modeled by labeled petri nets," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 2, pp. 1061–1069, April 2020.

[27] J. Pan and S. Hashtrudi-Zad, "Diagnosability Test for Timed Discrete-Event Systems," in *18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, Nov. 2006, pp. 63–72.

[28] P. Bouyer, F. Chevalier, and D. D'Souza, "Fault Diagnosis Using Timed Automata," in *Foundations of Software Science and Computational Structures*, ser. Lecture Notes in Computer Science, V. Sassone, Ed. Springer Berlin Heidelberg, 2005, pp. 219–233.

[29] P. Bonhomme, "Towards a diagnosability technique of P-time Petri nets systems," in *International Conference on Control, Decision and Information Technologies (CoDIT)*, Apr. 2016, pp. 018–023.

[30] F. Basile, M. P. Cabasino, and C. Seatzu, "Diagnosability Analysis of Labeled Time Petri Net Systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 3, pp. 1384–1396, Mar. 2017.

[31] ——, "State Estimation and Fault Diagnosis of Labeled Time Petri Net Systems With Unobservable Transitions," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 997–1009, Apr. 2015.

[32] Shengbing Jiang and R. Kumar, "Failure diagnosis of discrete-event systems with linear-time temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 49, no. 6, pp. 934–945, Jun. 2004.

[33] L. Ye and P. Dague, "An Optimized Algorithm of General Distributed Diagnosability Analysis for Modular Structures," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1768–1780, Apr. 2017.

[34] G. Lamperti and X. Zhao, "Diagnosis of Active Systems by Semantic Patterns," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 8, pp. 1028–1043, Aug. 2014.

[35] P. M. Merlin and D. J. Farber, "Recoverability of Communication Protocols - Implications of a Theoretical Study," *IEEE Transactions on Communications*, vol. 24, no. 9, pp. 1036–1043, Sep. 1976.

[36] Y. Pencolé, D. Kamenetsky, and A. Schumann, "Towards low-cost fault diagnosis in large component-based systems," in *IFAC Proceedings Volumes*, ser. 6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, vol. 39, Jan. 2006, pp. 1473–1478.

[37] B. Berthomieu, F. Peres, and F. Vernadat, "Bridging the gap between timed automata and bounded time Petri nets," in *4th International Conference Formal Modeling and Analysis of Timed Systems*, Paris, France, 9 2006, pp. 82–97.

[38] M. Hack, "Petri net language," Massachusetts Institute of Technology, Cambridge, MA, USA, Tech. Rep., 1976.

[39] B. Berthomieu, F. Peres, and F. Vernadat, "Model Checking Bounded Prioritized Time Petri Nets," in *Automated Technology for Verification and Analysis*, ser. Lecture Notes in Computer Science, K. S. Namjoshi, T. Yoneda, T. Higashino, and Y. Okamura, Eds. Springer Berlin Heidelberg, 2007, pp. 523–532.

[40] P. Wolper, M. Y. Vardi, and A. P. Sistla, "Reasoning about infinite computation paths," in *24th Annual Symposium on Foundations of Computer Science (sfcs 1983)*, Nov. 1983, pp. 185–194.

[41] É. Lubat, S. Dal Zilio, D. Le Botlan, Y. Pencolé, and A. Subias, "A State Class Construction for Computing the Intersection of Time Petri Nets Languages," in *17th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS)*, ser. LNCS, vol. 11750. Amsterdam, Netherlands: Springer, Aug. 2019.

Yannick Pencolé is a CNRS researcher at LAAS since 2006 and is the leader of the DISCO research team (Diagnosis and Supervisory Control). From 2003-2006, he was a researcher at ANU, Canberra, Australia. He received a PhD degree in 2002 on Model-based Diagnosis from the University of Rennes, France and his Habilitation à diriger des recherches from the university of Toulouse, France in 2018. His research interests include diagnosis of discrete-event systems and logical systems.

Audine Subias is Associate Professor in control and discete event systems at the National School of Applied Sciences (INSA) of Toulouse, working at CNRS-LAAS in the Diagnosis and Supervisory Control (DISCO) research team. She received the Habilitation à Diriger des Recherches in 2006 from the University of Toulouse, France. Her research interests include diagnosis of dynamic systems and diagnosability analysis, Monitoring and Health management.