Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Diagnosis of Labeled Time Petri Nets Using Time Interval Splitting

Baisi Liu [*,**] **Mohamed Ghazel** [*,***] **Armand Toguyéni** [*,**]

[*] *Univ. Lille Nord de France, F-59000, Lille, France*
[**] *École Centrale de Lille, F-59651, Villeneuve d'Ascq, France*
[***] *IFSTTAR, Cosys/Estas, F-59666, Villeneuve d'Ascq, France*
{*baisi.liu, armand.toguyeni*}*@ec-lille.fr, mohamed.ghazel@ifsttar.fr*

Abstract: This paper deals with fault diagnosis of timed discrete event systems (TDESs), using a nondeterministic model named labeled time Petri net (LTPN). Thanks to a skillful splitting of time intervals assigned to the LTPN transitions, analyzing diagnosability in such a timed context can be performed using techniques from the untimed context. Moreover, a deterministic structure called augmented state class set graph (ASG) is built on the fly, for both analyzing (Δ-)diagnosability and deriving an online diagnoser.

Keywords: fault diagnosis, discrete event system, labeled time Petri net, time interval splitting, on-the-fly analysis

1. INTRODUCTION

Fault diagnosis on discrete event systems (DESs) has been widely investigated and applied for high level analysis, since most industrial systems can be abstracted as a DES model to a certain degree (Lin, 1994).

DES diagnosis was first studied in the untimed context, where system behavior is described by logical sequences of events without time quantification. The pioneering work (Sampath et al., 1995) gives necessary and sufficient conditions for diagnosability (the ability to ensure that any fault can be diagnosed in a finite delay), and provides an approach for online diagnosis based on a structure called diagnoser automaton. However, it is difficult to apply this technique in the case of complex systems because they are not really scalable. This leads some authors to develop approaches based on time that allow us to use more expressive formalisms such as timed automata (TA) (Tripakis, 2002; Bouyer et al., 2005; Cassez and Tripakis, 2008). Other authors propose approaches based on formalisms such as time Petri nets (TPNs) that are more powerful in terms of expression capacity (Ghazel et al., 2009; Boel and Jiroveanu, 2013; Liu et al., 2013).

Our study is in the context of event-based diagnosis: the event set Σ is partitioned into two sets, $\Sigma = \Sigma_o \uplus \Sigma_u$, where Σ_o is a finite set of observable events and Σ_u is a finite set of unobservable events. Fault events are unobservable, i.e., $\Sigma_f \subseteq \Sigma_u$. Likewise, the set of transitions T, according to the mapping between Σ and T, is partitioned into the sets of observable and unobservable transitions: $T = T_o \uplus T_u$. Moreover, the faulty transitions are unobservable ($T_f \subseteq T_u$). Also, when various fault types are dealt with, the set of fault events is partitioned into m sets: $\Sigma_f = \biguplus_{i=1}^{m} \Sigma_{F_i}$, where Σ_{F_i} denotes one class of faults.

This study is an extension of the results in (Liu et al., 2013), to discuss the fault diagnosis problem of TDESs using a nondeterministic model, namely LTPN. The main contributions in this paper are as follows:

(1) An appropriate splitting of transitions time interval in such a way as to make explicit the differentiation of various system executions according to both the generated traces and the occurrence delays of observable events. As a consequence, diagnosability analysis can be advantageously brought to an untimed-like context, and existing results from works on DES-based untimed diagnosis can be used.
(2) A structure called ASG, which carries both reachability and fault propagation information, is built on the fly. Although LTPNs can be transformed into language-equivalent TA (Bérard et al., 2013), our on-the-fly approach is different from the TA-based approaches, which perform analysis on the basis of an existing complete TA model, whereas we rather compute the state space on the fly. The on-the-fly technique helps tackling the state explosion problem to a certain degree since, generally, a partial enumeration of the reachable states can be sufficient for analyzing diagnosability offline, and (if the system is diagnosable) emitting diagnosis verdicts online, without using an exhaustive diagnoser.

2. PREMILINARIES

2.1 Labeled Time Petri Net

A *TPN* is a modeling notation of TDESs. By associating with each transition an event, a TPN can be redefined as

⋆ The present research work has been partially supported by the International Campus on Safety and Intermodality in Transportation, the Nord-Pas-de-Calais Region, the European Community, the Regional Delegation for Research and Technology, the Ministry of Higher Education and Research, and the National Center for Scientific Research. The authors gratefully acknowledge the support of these institutions.

an LTPN such that each transition firing simultaneously produces the corresponding event.

Definition 1. (Bérard et al., 2005) An *LTPN* is a tuple $(P, T, \Sigma, Pre, Post, M_0, \varphi, SIM)$, where: P is a finite set of places; T is a finite set of transitions; Σ is a finite set of events; $Pre : P \times T \to \mathbb{N}$ and $Post : P \times T \to \mathbb{N}$ are the pre- and post-incidence mappings; $M_0 \in \mathbb{N}^{|P|}$ is the initial marking; $\varphi: T \to \Sigma$ is the labeling function; $SIM: T \to \mathbb{Q}_{\geq 0} \times (\mathbb{Q}_{\geq 0} \cup \{+\infty\})$ associates with each transition a *static interval mapping*, where $\mathbb{Q}_{\geq 0}$ is the set of non-negative rational numbers.

As shown in Figure 1(a), each LTPN transition is labeled with an event in Σ and an event can be assigned to different transitions. A state change of LTPN can be driven either by the firing of some transition or by time elapsing. Definitions of *state, state class (SC)* and their corresponding transitions are the same as in TPNs. Given two SCs C and C', C' is said to be *reachable* from C, if C' can be obtained by firing a sequence $\sigma \in T^*$, and we denote it by $C \xrightarrow{\sigma} C'$. An LTPN is *bounded* iff it has a finite number of SCs. For more details about the SC technique, the reader can refer to (Berthomieu and Diaz, 1991).

2.2 LTPN Language

A *dated firing sequence (DFS)* (Diaz, 2001) is a pair (σ, u), where $\sigma \in T^*$ is a possible firing sequence, and u is the sequence of firing dates of the transitions in σ. The set of achievable DFSs is denoted by $\mathcal{D}$.

Given a sequence of transitions (or events, dates) w, we denote by w^j the j^{th} element in w, and $|w|$ the length (number of elements) of w. For $a \in \Sigma$ and $w \in \Sigma^*$, we write $a \in w$ if there exists j such that $w^j = a$. We also write $w = w_1 w_2 \cdots w_n$ to say that w is the concatenation of $w_1, w_2, \ldots, w_n$, where $w_1, w_2, \ldots, w_n$ are sequences of transitions (or events, dates).

Definition 2. A *labeled dated firing sequence* (LDFS) of DFS (σ, u) is defined by (s, u), where $s = \varphi(\sigma)$, and φ is the extended form of the labeling function $\varphi : T^* \to \Sigma^*$.

Definition 3. The language generated by LTPN G is defined by $L(G) = \{(\varphi(\sigma), u) \mid (\sigma, u) \in \mathcal{D}\}$.

Here we use $\mathcal{D}_l$ to denote the set of LDFSs. With a slight abuse of notation, we shall write L instead of $L(G)$.

2.3 Diagnosability of LTPN

Given an LDFS p and a set of observable events Σ_o, let $P_o(p)$ be the LDFS obtained by erasing from p all the unobservable events and by summing up the relative delays to the delay of the immediate following observable event. Define the inverse projection operator P_o^{-1} as $P_o^{-1}(r) = \{p \in L \mid P_o(p) = r\}$ for $r \in (\Sigma_o \times \mathbb{Q}_{\geq 0})^*$. Given a language $L \subseteq \mathcal{D}_l$ and a string $p \in L$, the post-language of L after p denoted by L/p, is the language $L/p = \{r \in \mathcal{D}_l \mid pr \in L\}$.

In this paper, we will discuss the diagnosability in the framework of LTPNs. Without loss of generality, we consider only one class of faults.

Definition 4. Given an LTPN G, we say G is *diagnosable* if $\exists\ \Delta \in \mathbb{Q}_{\geq 0}$ such that $\forall\ (s, u) \in L$, if $s^{|s|} \in \Sigma_f$, $s^j \notin \Sigma_f$ for $j < |s|$ and $\forall\ (w, z) \in L/(s, u)$, $\sum_{j=1}^{|z|} z^j \geq \Delta$, then the following holds: $r \in P_o^{-1}(P_o(sw, uz)) \Rightarrow \exists\ e \in \Sigma_f$ s.t. $e \in r$ We also say here that G is Δ-*diagnosable*.

In simple terms, any fault in a diagnosable LTPN can be diagnosed within a finite delay upon its occurrence. Like for diagnosability of TA discussed in (Tripakis, 2002), for an LTPN there exists Δ_{min} such that, G is Δ-diagnosable for any $\Delta \geq \Delta_{min}$, and G is not Δ-diagnosable for any $\Delta < \Delta_{min}$. Looking for the Δ_{min} of a diagnosable LTPN is an interesting issue of practical significance since, in practice, we wish that the fault can be diagnosed as soon as possible and it is important to determine the minimum delay in which we ensure a fault can be diagnosed.

Before discussing the diagnosability of LTPNs, we make the following assumptions:

- The LTPN is bounded;
- No achievable cycle of unobservable transitions exists;
- Faults are permanent, i.e., when a fault occurs the system remains infinitely faulty.

Here we make similar assumptions to those for diagnosability analysis in the untimed context, since we will analyze the diagnosability of timed models using untimed analytical techniques. Note that the liveness condition is relaxed.

3. REACHABILITY ANALYSIS FOR LABELED TIME PETRI NETS WITH FAULT INFORMATION

In this section, we will first develop a structure which carries both the reachability relation between SCs and their corresponding fault information. This structure, called *augmented state class graph*, will be the basis of the diagnosability analysis of LTPNs in the sequel.

3.1 Augmented State Class (ASC)

Definition 5. An *ASC* is a pair $x = (C, y)$, which is associated to an achievable firing sequence $\sigma \in T^*$ such that $C_0 \xrightarrow{\sigma} C$, and y is computed by:

$$y = \begin{cases} F & \text{if } \exists\ j, \sigma^j \in T_f \\ N & \text{otherwise} \end{cases}$$

where SC C is reachable from the initial SC C_0 upon σ.

The initial ASC is defined by $x_0 = (C_0, N)$, as we consider that there is no fault in the system initially. Two ASCs $x = (C, y)$ and $x' = (C', y')$ are equivalent, iff $C = C'$, i.e., C and C' have the same marking and the same firing domain (Berthomieu and Diaz, 1991), and $y = y'$.

Let $\mathcal{N}$ be the set of ASCs relative to a given LTPN, mapping $\zeta : \mathcal{N} \times T^* \to \mathcal{N}$ defines transitions between ASCs. We say an ASC $x' = (C', y')$ is reachable from $x = (C, y)$ by $\sigma \in T^*$, denoted by $x \xrightarrow{\sigma} x'$, iff $C \xrightarrow{\sigma} C'$ and

$$y' = \begin{cases} F & \text{if } (y = F) \vee (\exists\ k, \sigma^k \in T_f) \\ N & \text{otherwise} \end{cases}$$

Consequently, the number of ASCs is at most twice the number of SCs. Thus, a bounded LTPN has a finite number of ASCs.

Given an ASC x, we shall call a *candidate sequence* of x any sequence of transitions $\sigma \in T_u^* T_o$ which is achievable starting from x. We denote by $Can(x)$ the set of candidate sequences of x. To each candidate sequence, one assigns a

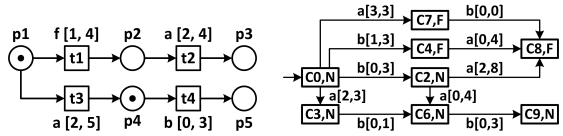
relative framing to its duration (Ghazel et al., 2009). This obtained interval contains all the possible firing dates of transition t relatively to x. We use the notation $SD(\sigma)$ to denote the sequence duration of a sequence σ.

3.2 Augmented State Class Graph (ASC-graph)

An *ASC-graph*, obtained through an ϵ-reduction on the SC graph, is a directed graph $(\mathcal{X}, \mathcal{A}, \gamma, x_0)$, where:

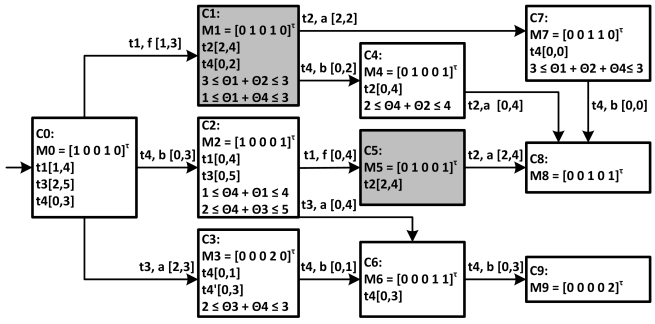
- $\mathcal{X} \subseteq \mathcal{N}$ is the set of ASC-graph nodes;
- $x_0 = (C_0, N)$ is the initial ASC-graph node;
- $\gamma : \mathcal{X} \times \Sigma_o \to 2^{\mathcal{X}}$ is the transition mapping between nodes. Given $x \in \mathcal{X}, e \in \Sigma_o$, $\gamma(x,e) = \{x' \mid \exists\ \sigma \in Can(x), \varphi(\sigma) = e, x \xrightarrow{\sigma} x'\}$;
- $\mathcal{A} \subseteq \mathcal{X} \times \Sigma_o \times \mathcal{I} \times \mathcal{X}$ is the set of directed arcs of the ASC-graph: $\mathcal{A} = \{(x, e, i, x') \mid \exists\ \sigma \in Can(x), x' \in \gamma(x,e),$ s.t. $SD(\sigma) = i, \varphi(\sigma) = e\}$, where $\mathcal{I}$ denotes the set of time intervals.

Example 6. Consider the LTPN G in Figure 1(a), where $T_u = T_f = \{t_1\}$ and $T_o = \{t_2, t_3, t_4\}$. The SC graph is given in Figure 1(c), where the grey boxes indicate SCs reached immediately after an observable transition.



(a) LTPN G

(b) The ASC-graph of G



(c) The SC graph of G

Fig. 1. The figures of Example 6

4. CHECKING DIAGNOSABILITY

Without loss of generality, we first discuss the diagnosability for one class of faults Σ_F. The generalization of our approach can be obtained just by repeating the same process for each class Σ_{F_i}. First, let us look at the time interval splitting technique.

4.1 Basic Interval Set (BIS)

The goal behind splitting a given finite set of time intervals is to generate a new set of split intervals fulfilling some constraints, that we call *BIS*. For this aim, we first introduce some basic notations on time intervals that will be used afterward.

Definition 7. A *left semi-interval* is a left-open interval:

- $a[= \{x \in \mathbb{Q}_{\geq 0} \mid x < a\} = [0, a[$ with $a \in \mathbb{Q}_{\geq 0}$, or
- $+\infty[= \{x \in \mathbb{Q}_{\geq 0}\} = [0, +\infty[$, or
- $a] = \{x \in \mathbb{Q}_{\geq 0} \mid x \leq a\} = [0, a]$;

a *right semi-interval* is a right-open interval, defined by

- $]a = \{x \in \mathbb{Q}_{\geq 0} \mid x > a\} =]a, +\infty[$ with $a \in \mathbb{Q}_{\geq 0}$, or
- $[a = \{x \in \mathbb{Q}_{\geq 0} \mid x \geq a\} = [a, +\infty[$.

Given an interval i, the corresponding left (resp. right) semi-interval is denoted by $l(i)$ (resp. $r(i)$), and the complementary set of semi-interval α is denoted by $\overline{\alpha}$. For $\beta = a]$ (resp. $a[;]a; [a)$, we denote $bound(\beta) = a$ and $border(\beta) =]$ (resp. $[;]; [$).

For two left (or two right) semi-intervals α and β, we say $\alpha = \beta$, if $bound(\alpha) = bound(\beta)$ and $border(\alpha) = border(\beta)$.

We define an order relation "$\prec$" between semi-intervals by:

- $\alpha \prec \beta$, if $bound(\alpha) < bound(\beta)$;
- $c[\prec [c \prec c] \prec]c$, for $c \in \mathbb{Q}_{\geq 0}$;
- $\alpha \prec +\infty[$, if $\alpha \neq +\infty[$.

The objective of defining this order relation between semi-intervals is to reorganize a set of semi-intervals for further computing basic interval sets (cf. Lines 3 and 7 in Algorithm 1). In order to eliminate nondeterminism in LTPNs, time interval splitting techniques will be developed to reassign each observable event with an interval in the BIS, such that each firing of an observable event with its relative time brings the system to a unique minimal ASC-set.

Definition 8. Given a finite time interval set A, the *BIS* of A, denoted by $BIS(A)$, is a set of disjoint non-empty time intervals β_j subject to:

- $\forall\ k \neq j, \beta_k \cap \beta_j = \emptyset$;
- $\forall\ \alpha \in A, \exists\ \beta_1, \beta_2, \cdots, \beta_m \in BIS(A)$ such that $\alpha = \bigcup_{j=1}^{m} \beta_j$;
- $\forall\ \beta_1, \beta_2 \in BIS(A), \beta_1 \neq \beta_2, \exists\ \alpha \in A$ such that $\beta_1 \cap \alpha = \emptyset, \beta_2 \cap \alpha \neq \emptyset$.

Here we emphasize that, for any finite interval set A, $BIS(A)$ has been proved to be finite and unique (Liu, 2014). Then the BIS of a finite interval set can be computed by Algorithm 1.

Algorithm 1 Computation of BIS

1: **Input:** A; ▷ A is a finite interval set.
2: **Output:** B; ▷ $B = BIS(A)$
3: $C \leftarrow \emptyset$; ▷ C is a list of semi-intervals ordered according to $\prec$.
4: **for all** $\alpha \in A$ **do** $C \leftarrow C \cup \{l(\alpha)\} \cup \{r(\alpha)\}$;
5: reorder C according to $\prec$;
6: $c_0 \leftarrow \overline{c_1}$; ▷ c_j $(j = 1, 2, \ldots)$ denotes, in the order of $\prec$, the j^{th} element of C.
7: $C \leftarrow C \cup \{c_0\}$;
8: **for** j from 1 to $(|C| - 1)$ **do**
9: **if** c_{j-1} is a right semi-interval **then** $\alpha \leftarrow c_{j-1}$;
10: **else** $\alpha \leftarrow \overline{c_{j-1}}$;
11: **if** c_j is a left semi-interval **then** $\beta \leftarrow c_j$;
12: **else** $\beta \leftarrow \overline{c_j}$;
13: $B \leftarrow B \cup \{(\alpha \cap \beta)\}$;
14: **return** B;

4.2 Augmented State Class Set (ASC-set)

In order to determinize an LTPN for state estimation and diagnosability analysis, we will gather the states reached by the same timed observation (observable event and its occurrence date) in some sets called ASC-sets.

An ASC-set is then an element of $2^{\mathcal{X}}$. The initial ASC-set is defined by $\{x_0\}$.

Given an ASC-set g, we say $e \in \Sigma_o$ is a *candidate event* of g, if $\exists\ x \in g, \gamma(x,e) \neq \emptyset$. We denote by $CES(g)$ the *candidate event set* of g.

The *candidate interval set (CIS)* of g relative to e is defined by $CIS(g,e) = BIS(Y)$, where $Y = \{SD(\sigma) \mid \exists\ x \in g, \sigma \in Can(x), \text{s.t. } \varphi(\sigma) = e\}$. In other words, $CIS(g,e)$ is the BIS relative to the intervals corresponding to the possible delays for e to occur from an element in g.

Let $\mathcal{G}$ be the set of reachable ASC-sets. Given $g \in \mathcal{G}$, $e \in CES(g)$ and $i \in CIS(g,e)$, the transition mapping between ASC-sets $\xi : \mathcal{G} \times \Sigma_o \times \mathcal{I} \to \mathcal{G}$ is defined by: $\xi(g,e,i) = \{x' \mid \exists\ x \in g, \sigma \in Can(x), \varphi(\sigma) = e, x \xrightarrow{\sigma} x' \text{ s.t. } i \subseteq SD(\sigma)\}$.

The ASC-set g is said to be

- normal, if $\forall\ (C,y) \in g, y = N$;
- F-certain, if $\forall\ (C,y) \in g, y = F$;
- F-uncertain, otherwise.

We denote $tag(g) = N$ (resp. F, U), if g is normal (resp. F-certain, F-uncertain).

4.3 Augmented State Class Set Graph (ASG)

The ASG is a deterministic digraph which will serve as a basis to check diagnosability. Here the term "deterministic" means that, given an ASC-set, upon the occurrence of an observable event, we can deduce with certainty which minimal candidate ASC-set the system will be possibly in. This will be ensured while discriminating between the candidate sequences using their sequence duration.

The *ASG* is a digraph $(\mathcal{G}, \mathcal{R}, \xi, g_0)$, where:

- $\mathcal{G} \subseteq 2^{\mathcal{X}}$ is the set of ASG nodes;
- $g_0 = \{x_0\} = \{(C_0, N)\}$ is the initial node;
- ξ is the transition mapping between ASCs;
- $\mathcal{R} \subseteq \mathcal{G} \times \Sigma_o \times \mathcal{I} \times \mathcal{G}$ is the set of ASG arcs: $\mathcal{R} = \{(g,e,i,g') \mid g' = \xi(g,e,i)\}$.

The ASG can be computed by Algorithm 2. It is worth noticing that in our technique the ASG is built on-the-fly from the SC graph.

Algorithm 2 Construction of the ASG

Input: the ASC-graph $(\mathcal{X}, \mathcal{A}, \gamma, x_0)$;
Output: the ASG;
$g_0 \leftarrow \{x_0\}$; ▷ initialization
$\mathcal{G}_{con} \leftarrow \{g_0\}$; ▷ $\mathcal{G}_{con}$ is the set of ASCs to be considered.
$\mathcal{G}_{vst} \leftarrow \emptyset$; ▷ $\mathcal{G}_{vst}$ is the set of ASCs that have been considered.
while $\mathcal{G}_{con} \neq \emptyset$ **do**
  pick a node $g \in \mathcal{G}_{con}$ s.t. $g \notin \mathcal{G}_{vst}$;
  for all $e \in CES(g)$ **do**
    $Y \leftarrow CIS(g,e)$;
    for all $i \in Y$ **do**
      $\mathcal{G}_{con} \leftarrow \mathcal{G}_{con} \cup \{\xi(g,e,i)\}$;
  $\mathcal{G}_{con} \leftarrow \mathcal{G}_{con} \backslash \{g\}$;
  $\mathcal{G}_{vst} \leftarrow \mathcal{G}_{vst} \cup \{g\}$;

An ASG node g' is reachable from g, if there is a path from g to g' in the ASG. We write $succ(g)$ to denote all the successor (or reachable) nodes from g. Therefore, given an ASG $(\mathcal{G}, \mathcal{R}, \xi, g_0)$, $\mathcal{G} = succ(g_0)$.

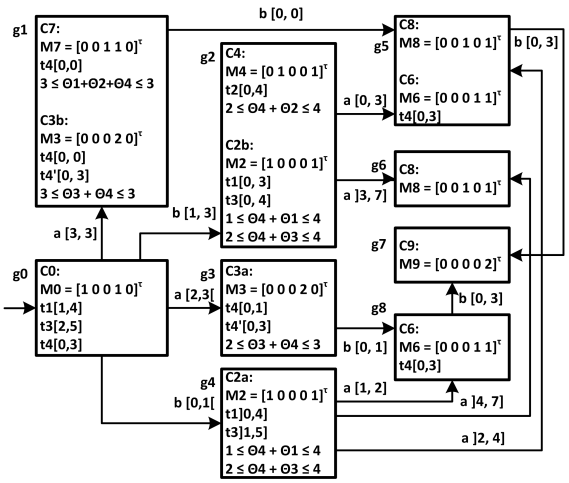


Fig. 2. The ASG of the LTPN in Figure 1(a)

4.4 Conditions for Undiagnosability

As explained earlier, the ASG offers a state representation that distinguishes between reachable states based on an explicit discrimination taking into account both observable events and their possible occurrence dates. Defining such a structure makes it possible to use similar analysis as in the untimed context. However, the following considerations related to time still need to be taken into account.

Condition 1: indeterminate cycle

Recall that the necessary and sufficient condition for the diagnosability of an automaton is the absence of an indeterminate cycle as proved in (Sampath et al., 1995). We can extend this condition for the diagnosability analysis of LTPN based on the ASG, since the time interval splitting technique makes it possible to derive an untimed-diagnoser-like structure, by making explicit the distinction between sequences on the basis of temporal criteria in the ASG model structure.

By analogy with the untimed context, we define an indeterminate cycle in an ASG as a cycle composed of finite nodes in the graph, such that for any node g in this cycle, two ASCs $x_1, x_2 \in g$ exist, x_1 is a faulty ASC in a cycle composed of faulty nodes in the ASC-graph, while x_2 is a normal ASC in a cycle composed of normal nodes.

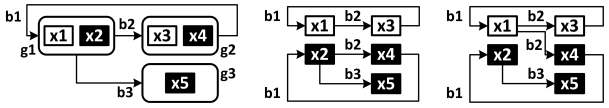
Proposition 9. The LTPN is undiagnosable if an indeterminate cycle in the ASG exists.

This is obvious according to the explanation of indeterminate cycle. Note that a cycle of F-uncertain ASCs in the ASG (Figure 3(a), where the black boxes are faulty ASCs and the white ones are normal) is not necessarily an indeterminate cycle. If this cycle corresponds to two ASC cycles in the ASC-graph such that one is a normal cycle (x_1, x_3) and the other is a faulty one (x_2, x_4) as in Figure 3(b), then g_1 and g_2 form an indeterminate cycle. Otherwise, they do not (Figure 3(c)).

Condition 2: infinite sequence duration in certain cases

We define mapping $fdelay : \mathcal{G} \to \mathbb{Q}_{\geq 0} \cup \{+\infty\}$ as follows:

- If $tag(g) = N$, then $fdelay(g) = 0$.
- Otherwise, if g' is the predecessor of g, and



(a) (b) (c)

Fig. 3. Illustration of indeterminate cycle

- If $tag(g') = N$, i.e., $\exists\ x' \in g', x \in g, \sigma' \in (T_u \backslash T_f)^*, t_f \in T_f, \sigma \in T_u^* T_o$, such that $x' \xrightarrow{\sigma' t_f \sigma} x$, then $fdelay(g) = max\{SD(\sigma)\}$, or
- If $tag(g') \neq N$, i.e., $\exists\ x' \in g', x \in g, \sigma \in T_u^* T_o$, such that $x' \xrightarrow{\sigma} x$, then $fdelay(g) = fdelay(g') + max\{SD(\sigma)\}$.

During the on-the-fly building of the ASG, whenever g is equal to an existing node g'', both $fdelay(g)$ and $fdelay(g'')$ should be updated by $max\{fdelay(g), fdelay(g'')\}$. In other terms, $fdelay(g)$ always records the maximum time delay between the first occurrence of fault and g, which will be used to determine the undiagnosability or compute (if the LTPN is diagnosable) Δ_{min}.

Proposition 10. An LTPN is undiagnosable if $\exists\ g \in \mathcal{G}$ s.t. $fdelay(g) = +\infty$, while the ASG is built on the fly.

Proof. According to the definition of mapping *delay*, $delay(g) = +\infty$ means that a possible occurred fault cannot be diagnosed in a finite delay after its occurrence, thus the LTPN is undiagnosable. □

Condition 3: dead subset in certain cases

Note that here we also deal with non-live systems. For this, let us introduce the following definitions. An ASC-set g is:

- *undead or non-blocking*, if $\forall\ x \in g, \exists\ t \in T$ s.t. $x \xrightarrow{t}$;
- *dead*, if $\forall\ x \in g, \nexists\ t \in T$ s.t. $x \xrightarrow{t}$;
- *quasi-dead*, otherwise.

Given a quasi-dead ASC-set g, we define the *dead subset* of g as the set of all dead ASCs in g, which can be formalized as: $DS(g) = \{x \in g \mid \nexists t \in T \text{ s.t. } x \xrightarrow{t}\}$.

We will now discuss some conditions for undiagnosability w.r.t the liveness of ASCs.

Proposition 11. An LTPN is undiagnosable if a quasi-dead ASC-set g exists, such that

(3) $DS(g)$ is F-uncertain, or
(4) $DS(g)$ is F-certain and a normal successor ASC-set g' exists, such that g' may be reached upon an infinite delay $(+\infty)$.

Proof. For (3), an F-uncertain dead subset means that some ASCs in this set may be reachable by firing a sequence containing a fault, while others can be reached without any fault having occurred. Furthermore, it is impossible to distinguish them by further observation, since they are all dead and the system will remain in F-uncertain state forever. For (4), if g' is reachable upon an infinite delay, one cannot determine whether the system is blocked in the (faulty) dead subset of g ($DS(g)$), or it is still on the way to g', which means that it is possible that no fault has occurred in the state g' in a finite delay after the fault, i.e., we do not know if a fault has occurred. □

4.5 On-the-Fly Checking of Diagnosability

Proposition 12. A bounded LTPN is diagnosable *iff* none of the conditions in Propositions 9, 10 and 11 holds.

Proof. $(\Rightarrow)$: This condition is proposed from three perspectives that we consider:

(1) With the help of splitting time intervals, the behavior of LTPN is characterized as in the untimed context, where the absence of indeterminate cycle has been proved to be necessary and sufficient condition for diagnosability (Sampath et al., 1995).
(2) This is the restriction from the definition of diagnosability of LTPN.
(3) This is the restriction from the perspective of considering non-live TDESs.

$(\Leftarrow)$: The negation of these three conditions have been proved to be necessary by Propositions 9, 10 and 11, since each of the conditions in Propositions 9, 10 and 11 is sufficient for undiagnosability. □

We have shown that diagnosability can be checked while building ASG. Actually, building the whole ASG would be similar to the approach based on state enumeration, often consuming much memory while dealing with large systems, even if this burdensome work could be performed offline. Yet, there is still a difference w.r.t this approach, since ASG branch building is stopped as soon as an F-certain ASC-set is found or if one of the conditions for undiagnosability (cf. Propositions 9, 10 and 11) holds. In order to tackle this problem, we will propose a new approach to checking diagnosability on the basis of on-the-fly building of the ASG, as shown in Algorithm 3. Moreover, we determine the minimum value Δ_{min} for which the system is diagnosable. Hence, when the system is diagnosable and with Δ_{min} being determined, the system is Δ-diagnosable for any $\Delta \geq \Delta_{min}$ and is not Δ-diagnosable for any $\Delta < \Delta_{min}$. Note that this on-the-fly investigation of diagnosability is the adaptation of the techniques that we have developed for the untimed context (Liu et al., 2014).

5. ONLINE DIAGNOSIS OF LTPN

Let us discuss how online diagnosis for a diagnosable LTPN model is performed, using a deterministic structure called *labeled timed diagnoser (LTD)* that will be developed. By observing events with their corresponding occurrence dates online, one can deduce with certainty which state (Normal, F-uncertain or F-certain) the system may be in and give the verdict pertinent to fault occurrences.

The LTD is obtained from the ASG by erasing all the information except fault tags for each node in the ASG and observable events labeling the arcs with their corresponding intervals. This procedure deletes all the information unnecessary for online diagnosis.

As shown in Figure 5, for each F-uncertain quasi-dead node g, a virtual node g' labeled with "F" is created as a successor to g, and the arc from g to g' is labeled with (ϵ, i), where ϵ is an empty event indicating that no event is observed, i is the interval from the maximum firing date of the other firable transitions to $+\infty$. Note that, this virtual node does not belong to the LTD, it just helps to diagnose a fault when dealing with non-live systems.

Algorithm 3 On-the-fly building of ASG, checking diagnosability and computing Δ_{min}

Input: the ASC-graph;
Output: diagnosability of G and (if G is diagnosable) Δ_{min};
$g_0 \leftarrow \{x_0\}$;
$\Delta_{min} \leftarrow 0$;
$\mathcal{G}_{vst} \leftarrow \emptyset$;
$\mathcal{G}_{con} \leftarrow \{g_0\}$;
while $\mathcal{G}_{con} \neq \emptyset$ **do** pick a node $g \in \mathcal{G}_{con}$;
  for all $e \in CES(g, e)$ **do** $\mathcal{I} \leftarrow CIS(g, e)$;
    for all $i \in \mathcal{I}$ **do** $g' \leftarrow \xi(g, e, i)$;
      if $tag(g') = N$ **then**
        if g has a flag and $max(SD(i)) = +\infty$ **then**
          return G is undiagnosable;
        if $g' \notin \mathcal{G}_{vst}$ **then** $\mathcal{G}_{vst} \leftarrow \mathcal{G}_{vst} \cup \{g'\}$;
          $\mathcal{G}_{con} \leftarrow \mathcal{G}_{con} \cup \{g'\}$;
      if $tag(g') = U$ **then**
        if $fdelay(g') = +\infty$ **then return** G is undiagnosable;
        if $tag(DS(g')) = U$ **then return** G is undiagnosable;
        if $tag(DS(g')) = F$ **then** give g' a flag;
        if $\exists\ g'' \in \mathcal{G}_{vst}$ s.t. $g'' = g'$ **then**
          if g' is in an indeterminate cycle **then**
            return G is undiagnosable;
          else if $fdelay(g') > fdelay(g'')$ **then**
            $\Delta = fdelay(g') - fdelay(g'')$;
            UpdFdelay$(g'', \Delta, \Delta_{min})$;
        else $\mathcal{G}_{vst} \leftarrow \mathcal{G}_{vst} \cup \{g'\}$;
          $\mathcal{G}_{con} \leftarrow \mathcal{G}_{con} \cup \{g'\}$;
      if $tag(g') = F$ **then**
        if $fdelay(g') = +\infty$ **then return** G is undiagnosable;
        else $\Delta_{min} \leftarrow max(\Delta_{min}, fdelay(g'))$;
          $\mathcal{G}_{vst} \leftarrow \mathcal{G}_{vst} \cup \{g'\}$;
  $\mathcal{G}_{con} \leftarrow \mathcal{G}_{con} \backslash \{g\}$;
return G is Δ_{min}-diagnosable;
function UpdFdelay$(g, \Delta, \Delta_{min})$
  for all $z \in succ(g)$ **do**
    if $tag(z) \neq N$ **then** $fdelay(z) \leftarrow fdelay(g) + \Delta$;
      $\Delta_{min} \leftarrow max\{\Delta_{min}, fdelay(z)\}$;
    UpdFdelay$(z, \Delta, \Delta_{min})$;

The tags associated with each node in an LTD provide the same information as those in the ASG nodes:

- "N" means that no fault has occurred;
- "U" denotes that a fault has possibly occurred, and further observation is needed;
- "F" denotes that a fault has occurred with certainty.

Given a system behavior presented by a sequence of observable events with their corresponding event occurrence dates, one can find whether a fault has occurred or not, with the help of the LTD.

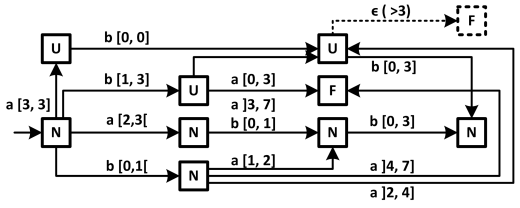


Fig. 4. Diagnoser of the LTPN in Figure 1(a)

6. CONCLUSION

This study shows that, compared with the untimed context, considerations on both logical and timing system behavior make it more challenging to discuss the state evolution and fault propagation. Thanks to the time interval splitting technique that we have developed, the timed diagnosis problem can be handled in a similar way as in the untimed context. The necessary and sufficient conditions for diagnosability of LTPN models are given on the basis of on-the-fly building of a dedicated structure called ASG, which carries the necessary pieces of information to check diagnosability. Furthermore, a diagnoser named LTD can be derived for online diagnosis in a straightforward way.

As future directions, we will consider using other notations to characterize the dynamic behavior of LTPNs, e.g., zone graph (Gardey et al., 2004), to simplify the reasoning process and improve the efficiency of the approach.

REFERENCES

Bérard, B., Cassez, F., Haddad, S., Lime, D., and Roux, O.H. (2013). The expressive power of time Petri nets. *Theoretical Computer Science*, 474, 1–20.

Bérard, B., Cassez, F., Haddad, S., Lime, D., and Roux, O.H. (2005). Comparison of different semantics for time Petri nets. In *Automated Technology for Verification and Analysis*, 293–307. Springer.

Berthomieu, B. and Diaz, M. (1991). Modeling and verification of time dependent systems using time Petri nets. *IEEE Transactions on Software Engineering*, 17(3), 259–273.

Boel, R.K. and Jiroveanu, G. (2013). The on-line diagnosis of time Petri nets. In *Control of Discrete-Event Systems*, chapter 17, 343–364. Springer.

Bouyer, P., Chevalier, F., and D'Souza, D. (2005). Fault diagnosis using timed automata. In *Foundations of Software Science and Computational Structures*, 219–233. Springer.

Cassez, F. and Tripakis, S. (2008). Fault diagnosis with static and dynamic observers. *Fundamenta Informaticae*, 88(4), 497–540.

Diaz, M. (2001). *Les réseaux de Petri - modèles fondamentaux*. Hermés, Paris.

Gardey, G., Roux, O., and Roux, O.H. (2004). Using zone graph method for computing the state space of a time Petri net. In *Formal modeling and analysis of timed systems*, 246–259. Springer.

Ghazel, M., Toguyéni, A., and Yim, P. (2009). State observer for DES under partial observation with time Petri nets. *Discrete Event Dynamic Systems*, 19(2), 137–165.

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

Liu, B. (2014). *An efficient approach for diagnosability and diagnosis of DES based on labeled Petri nets in untimed and timed contexts*. Ph.D. thesis, Univ. Lille Nord de France.

Liu, B., Ghazel, M., and Toguyéni, A. (2013). Évaluation à la volée de la diagnosticabilité des systèmes à événements discrets temporisés. *Journal Européen des Systèmes Automatisés, Édition spéciale MSR'13, Modélisation des Systèmes Réactifs*, 47(1-2-3), 227–242.

Liu, B., Ghazel, M., and Toguyéni, A. (2014). Toward an efficient approach for diagnosability analysis of DES modeled by labeled Petri nets. In *The 13th European Control Conference (ECC'14)*.

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

Tripakis, S. (2002). Fault diagnosis for timed automata. In *Formal Techniques in Real-Time and Fault-Tolerant Systems*, 205–221. Springer.