2013 European Control Conference (ECC)
July 17-19, 2013, Zürich, Switzerland.

Fault Diagnosis Graph of Time Petri Nets

Xu Wang, Cristian Mahulea and Manuel Silva

***Abstract*— This paper proposes an online approach for the fault diagnosis for time discrete event systems, which are modeled by time Petri net. The observation is given by a subset of transitions whose occurrence is always observable. Faults correspond to a subset of the transitions whose firing are not observable. According to the most of the literature on discrete event systems, we define three fault states, namely N, F and U, corresponding to normal, fault and uncertain states, respectively. The proposed approach use a *Fault Diagnosis Graph* (FDG). It is adapted from state class graph by keeping only the necessary information for computation of the fault states and removing the unnecessary states. Some algorithms to compute after each observation only part of the FDG required to update the diagnosis states are given.**

I. Introduction

The increasing of complexity of systems makes the efficiency of fault diagnosis to be critical. A fault is an abnormal behavior in a system, and diagnosis is the process to detect faults. In recent years contributions have been proposed in the literature to deal with fault detection and state estimation in discrete event systems, untimed and time systems [1], [2], [3]. The diagnosis of discrete event systems based on Petri nets has been an active research area in the last decades. Cabasino et al. [4] present a fault detection method for discrete event systems using Petri nets. Some transitions of the net are unobservable and some of them describe fault behaviors. The diagnosis is based on the notion of *basis marking* and the *basis reachability graph* for fault detection on untimed Petri net. The approach is applied to PN systems whose unobservable subnet is acyclic. In [5], integer linear programming problem (ILP) is applied on the diagnosis of systems modeled by time Petri nets. The constraints of the ILP characterize possible firing sequences according to the observation. The firing of fault transitions are detected by maximizing the firing counts of fault transitions. Because for each observation, some inequalities are added into the set of constraints, the number of constraints may become very large. Basile et al. [6] consider fault diagnosis on timed Petri net. Timed explanation trees are computed for diagnosis and the computation of the trees are based on ILPs.

In this paper, systems are modeled using time PN. In the net, the firings of some transitions are not observable. Some unobservable transitions describe regular behaviors, while others present fault behaviors. We assume the net structure, the initial marking and the time bounds of durations of all transitions are known. Our problem is to detect the firing of fault transitions by observing the firing of the observable transitions. The observation problem is solved on the FDG (Sec. IV). We present algorithms to construct this graph and to diagnose on it. The FDG is a directed graph in which vertices are reachable markings (states) obtained by firing observable transitions and edges are firing sequences. The graph is incrementally computed after the state class graph [7] of unobservable subnet is computed.

This work has been partially supported by CICYT - FEDER project DPI2010-20413. The Group of Discrete Event Systems Engineering (GISED) is partially co-financed by the Aragonese Government (Ref. T27) and the European Social Fund. The authors are with Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, María de Luna 1, E-50018 Zaragoza, Spain. Email:{xuwang, cmahulea, silva}@unizar.es

II. Time Petri net

In this section, we recall the basic definition of time Petri net system (for a general introduction, see [8], [9]).

Definition 1: A *time PN system* is a pair $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle$, where $\mathcal{N} = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post}, I \rangle$ is a net structure with a set of places P; a set of transitions T; the pre and post incidence matrices $\boldsymbol{Pre}, \boldsymbol{Post} \in \mathbb{N}_{\geq 0}^{|P| \times |T|}$; $I : T \to \mathbb{Q}_{\geq 0} \times \mathbb{Q}_{\geq 0} \cup \{\infty\}$ is the time function associating a *time interval* to each transition; and $\boldsymbol{m}_0 \in \mathbb{N}_{\geq 0}^{|P|}$ is the initial marking, where $|P|$ is the number of places and $|T|$ is the number of transitions.

For every node $v \in P \cup T$, the set of its input and output nodes are denoted as ${}^\bullet v$ and $v^\bullet$, respectively. A transition $t \in T$ is enabled at a marking $\boldsymbol{m}$ if and only if $\boldsymbol{m} \geq \boldsymbol{Pre}[\cdot, t]$. Considering $I(t) = [l, u]$, then when t is enabled, it cannot be fired earilier than l time unites and if u time unites have passed, it must be fired. If a marking $\boldsymbol{m}'$ is reachable from $\boldsymbol{m}$ by firing a sequence $\sigma = t_{i1} t_{i2} \cdots t_{in} \in T^*$, where $t_{ij} \in T, j = 1, 2, \ldots, n$; the fundamental state equation can be written as $\boldsymbol{m}' = \boldsymbol{m} + \boldsymbol{C} \cdot \vec{\sigma}$, where $\vec{\sigma} \in \mathbb{N}^{|T|}$ is the firing count vector of σ; $\boldsymbol{m}[\sigma\rangle$ denotes that σ is fireable from $\boldsymbol{m}$, while $\boldsymbol{m}[\sigma\rangle\boldsymbol{m}'$ means the firing of σ drives $\boldsymbol{m}$ to $\boldsymbol{m}'$. The single server semantic is used, which means that a transition cannot be enabled simultaneously more than once.

The set of transitions T is partitioned into two: T_o and T_u, where T_o is the set of *observable* transitions, whose firing can be detected by an external observer, and T_u is the set of *unobservable* transitions. The firing sequence σ^o is an observable firing sequence, if $t \in \sigma^o$, then $t \in T_o$; σ^u is an unobservable firing sequence, if $t \in \sigma^u$, then $t \in T_u$. An observation function $\lambda : \sigma \to T_o^*$, where T_o^* is the Kleene closure of T_o, extracts a sequence of observable transitions $\lambda(\sigma)$ from σ. Let $\sigma = \sigma_1^u \sigma_1^o \sigma_2^u \sigma_2^o \cdots \sigma_n^u$, then $\lambda(\sigma) = \sigma_1^o \sigma_2^o \cdots \sigma_{n-1}^o$. Observable transitions are represented as white rectangles, while unobservable ones as black rectangles. We use $|\sigma|$ to denote the number of transitions in σ.

Definition 2: The unobservable subnet of a time Petri net $\mathcal{N} = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post}, I \rangle$ is $\mathcal{N}_u = \langle P, T_u, \boldsymbol{Pre}_u, \boldsymbol{Post}_u, I_u \rangle$, where 1) P is the set of places, 2) T_u is

the set of unobservable transitions in $\mathcal{N}$, 3) $\boldsymbol{Pre}_u$ and $\boldsymbol{Post}_u$ are pre and post incidence matrices restricted to T_u, 4) $I_u : T_u \to \mathbb{Q}_0 \times \mathbb{Q}_0 \cup \{\infty\}$.

Definition 3: An observed word is a sequence of ordered pairs $w = \langle t_1, \tau_1\rangle \cdots \langle t_k, \tau_k\rangle \in (T_o \times \mathbb{Q}_0)^*$, in which t_1 is the first observed transition, while t_k is the latest observed transition at τ_k. Let $\langle \mathcal{N}, \boldsymbol{m}_0\rangle$ be a time Petri system and $w = \langle t_1, \tau_1\rangle \cdots \langle t_k, \tau_k\rangle$ be an observed word. We define the *set of firing sequences consistent with* w by $\mathcal{L}(w) = \{\sigma | \boldsymbol{m}_0[\sigma\rangle, w' = t_1 \dots t_k = \lambda(\sigma), t_i$ is fireable at $\tau_i, i = 1, \dots k\}$.

A *fault* is modeled by an unobservable transition, but there may be unobservable transitions whose firing corresponds to regular behaviors. To model fault and regular behaviors in Petri net, the set of unobservable transitions is partitioned into two subsets $T_u = T_f \cup T_{reg}$, where T_f includes all faulty transitions and T_{reg} includes all transitions relative to unobservable but regular events. The set T_f is further partitioned into r subsets as $T_f = T_f^1 \cup T_f^2 \cup \cdots \cup T_f^r$, where all transitions in the same subset correspond to the same fault class. We say that the i-th fault has occurred when a transition in T_f^i has fired.

Definition 4: A *diagnoser* is a function $\Delta : (T_o \times \mathbb{Q}_0)^* \times \{T_f^1, T_f^2, \dots, T_f^r\} \to \{\text{N, F, U}\}$, where $(T_o \times \mathbb{Q}_o)^*$ is the set of observed words $w = \langle t_1, \tau_1\rangle \cdots \langle t_j, \tau_j\rangle$. The diagnoser associates to each observed word w and to each fault class $T_f^i, i = 1, \dots, r$, a diagnosis state. a) $\Delta(w, T_f^i) =$ N if for all $\sigma \in \mathcal{L}(w)$ and for all $t_f \in T_f^i$ it holds $t_f \notin \sigma$. b) $\Delta(w, T_f^i) =$ U if a) there exists $\sigma \in \mathcal{L}(w)$ and $t_f \in T_f^i$ such that $t_f \in \sigma$, but b) there exists $\sigma' \in \mathcal{L}(w)$ and $\forall t_f \in T_f^i$ such that $t_f \notin \sigma'$. In this case, a fault transition of class i may have occurred or not, i.e., it is uncertain. c) $\Delta(w, T_f^i) =$ F if for all $\sigma \in \mathcal{L}(w)$ and for all $t_f \in T_f^i$ it holds $t_f \in \sigma$.

III. State Class Graph of time PN

In this section, we recall the algorithms from [9] for the computation of the State Class Graph (SCG). A *state class* is a pair $\alpha = \langle \boldsymbol{m}, F\rangle$, where $\boldsymbol{m}$ is a reachable marking and F is a conjunction of inequalities representing the firing domains, i.e., the possible firing intervals of transitions. If t_j is an enabled transition at $\boldsymbol{m}$ and has associated a firing interval $[l, u]$ then there exists an inequality of the form $l \le x_j \le u$ in F, where x_j is the time delay when t_j can fired at $\boldsymbol{m}$.

For the computation of the SCG the following two functions are required (for details, see [9]): 1) *isFireable*(α, t_j) returns true if t_j can be fired at α. In the algorithm, the set of enabled transitions at $\boldsymbol{m}$ is denoted by $\text{En}(\boldsymbol{m}) = \{t_j | \boldsymbol{m} \ge \boldsymbol{Pre}[\cdot, t_j]\}$. 2) *succ*$(\alpha, t_j)$ computes the successor of α by firing t_j.

Definition 5: [9] A *state class graph* is a triple $SCG = \langle C, \twoheadrightarrow, \alpha_1\rangle$, where 1) $\alpha_1 = \langle \boldsymbol{m}_0, F_0\rangle$ is the initial state, 2) $\alpha \overset{t}{\twoheadrightarrow} \alpha'$ is the set of edges, in which $\alpha \overset{t}{\twoheadrightarrow} \alpha'$ iff isFireable(α, t) and $\alpha' = \text{succ}(\alpha, t)$, 3) $C = \{\alpha | \alpha_1 \twoheadrightarrow^* \alpha\}$ is the set of reachable state classes, where $\twoheadrightarrow^*$ is the reflexive and transitive closure of $\twoheadrightarrow$.

In order to compute the state class graph, we propose Alg. 1 to compute the whole state class graph of a given Petri net system. Alg. 1 implements two additional steps 12 and 13 compared with the original version in [9]. The SCG is returned as a tuple $\langle C, \twoheadrightarrow, \alpha_1\rangle$, where C is the set of reachable state classes, $\twoheadrightarrow$ is the set of edges in SCG, and α_1 is the initial state class. An edge $\langle \alpha, t, \alpha'\rangle \in \twoheadrightarrow$ means that t is fireable at α and α' is obtained by firing t at α_1. The set W keeps the states which are not explored yet. For a newly computed state class α', if it has not been explored, it is added to W (step 10) and the edge is inserted into $\twoheadrightarrow$. Considering an edge $\langle \alpha, t, \alpha'\rangle$, if α' exists in C, but the edge does not exist in $\twoheadrightarrow$, then the relation will be added into $\twoheadrightarrow$, while α' is not inserted neither in C nor in W (steps 12 and 13).

Algorithm 1 $\langle C, \twoheadrightarrow, \alpha_1\rangle = \text{SCG}(\mathcal{N}, \boldsymbol{m}_0)$

1: $\alpha_1 := \langle \boldsymbol{m}_0, F_0\rangle$
2: $C := \{\alpha_1\}$
3: $\twoheadrightarrow := \emptyset$
4: $W := \{\alpha_1\}$
5: **while** $W \neq \emptyset$ **do**
6: get and remove $\alpha := \langle \boldsymbol{m}, F\rangle$ from W
7: **for each** $t \in \text{En}(\boldsymbol{m})$ s.t. isFireable(α, t) **do**
8: $\alpha' := \text{succ}(\alpha, t)$
9: **if** $\alpha' \notin C$ **then**
10: add α' to C and to W
11: add $\langle \alpha, t, \alpha'\rangle$ to $\twoheadrightarrow$
12: **else if** $\langle \alpha, t, \alpha'\rangle \notin \twoheadrightarrow$ **then**
13: add $\langle \alpha, t, \alpha'\rangle$ to $\twoheadrightarrow$
14: **end if**
15: **end for**
16: **end while**
17: **return** $\langle C, \twoheadrightarrow, \alpha_1\rangle$

In general, the computation of the SCG is time consuming due to the large number of reachable states. However, for the fault detection we do not need to compute the full SCG but only the subgraph corresponding to the actual observation. For example, when building the fault class according to the empty observation, i.e., no transition has been observed, what we need to compute is the SCG of unobservable subnet starting from the initial state, i.e., firing only unobservable transitions. In the following sections we propose some algorithms in order to compute the SCG containing only the information necessary to update the fault states.

IV. Fault Diagnosis on Time Petri Nets

A. Fault Diagnosis Graph

In order to detect the fault occurrences it is necessary to check all possible firing sequences consistent with the observation and see if the fault transitions belong or not to them. Therefore, the markings in which the net system may be is not playing the central role but the sequence firing. For this reason we construct the FDG by adding the sequence information to the SCG and removing the state classes that are not important.

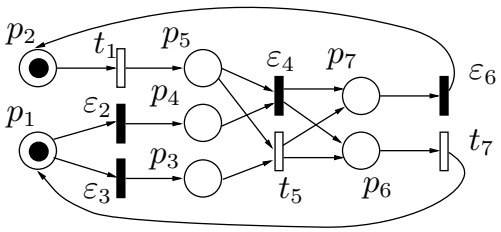
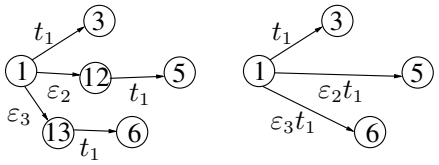


Fig. 1. A Petri net model where ε_4 is fault transition



(a) SCG (b) FDG

Fig. 2. SCG and FDG of the PN model in Fig. 1

Let us consider the PN system in Fig. 1 where t_1, t_5 and t_7 are observable transitions and the firing durations of all transitions are $[1, 4]$. We compute the part of the SCG corresponding to the firing of unobservable transitions and then t_1. By using Alg. 1, the SCG in Fig. 2(a) is obtained, where α_1 (node 1) is the initial state class. For the fault detection, it will be important to obtain the state classes corresponding to the firing of the observable transitions since after their firings the diagnosis state should be updated. For the above set of transitions, it is the firing of t_1. The firings of ε_2 and ε_3 should be considered only when the diagnosis state at α_1 is computed. Therefore, state classes α_{12} and α_{13} that are obtained by firing the unobservable transitions can be removed from the FDG and information regarding the unobservable transitions that have been fired will be added to the labels of edges. Details of nodes and edges of the FDG in Fig. 2(b) can be found in Sec. V.

For a compact representation, the firing count vector of the corresponding sequence is recorded, instead of firing sequences. However, the time interval of these sequences should be computed in order to explore the future nodes.

A FDG is a directed multigraph where an edge from a node α to a node α' is labeled with$\langle \vec{\sigma}, I_\sigma \rangle$, meaning that α' is reachable from α by firing a sequence $\sigma = \sigma_u t_j$, where σ_u is an unobservable firing sequence with $t_j \in T_o$, $\vec{\sigma}$ is the firing count vector of σ, while I_σ is its time interval.

B. Fault Diagnosis Algorithm

The fault diagnosis algorithm is presented as Alg. 2. Before any observation, i.e., the observation word is $w = \epsilon$, the SCG of the unobservable subnet with the initial state α_1 is computed (step 2). Based on it, the FDG corresponding to the empty word is obtained (step 3). In the same step, W is initialized with the set of vertices in the actual FDG minus α_1. The states in W will not be explored at this observation, because it is enough to check the diagnosis states based only on the output edges of α_1 of the actual FDG. If a transition in a fault class T_f^i is fired at all output edges of α_1, the diagnosis state is fault; if it is not fired in any output edge of α_1, the diagnosis state is normal; otherwise, the diagnosis state is uncertain (step 4). The set $\mathcal{S}$ contains vertices of FDG consistent with the observation and the paths from the initial state to them, i.e., if $\langle \alpha, \vec{\sigma}_\alpha \rangle \in \mathcal{S}$ and $w = t_1 t_2 \dots t_j$ is the observation, there exists a path $\alpha_1 \xrightarrow{t_1} \alpha_1 \xrightarrow{t_2} \cdots \xrightarrow{t_j} \alpha$ in the FDG. When a new observation comes (step 6), the FDG is updated if there exists unexplored vertices in the FDG (step 7). For each α in $\mathcal{S}$, if it has an output edge that, in the firing count vector, the firing count of the observed transition is 1 and the corresponding output state is not explored (step 8), then the unobservable SCG starting from the output state is computed and the FDG and W are updated accordingly (steps 9 to 11). After the computation of the FDG according to the current observation, the fault classes are diagnosed, while $\mathcal{S}$ is updated (step 14).

Algorithm 2 General fault diagnosis algorithm

1: $v := \epsilon$
2: $\mathcal{G}_{scg} := \text{SCG}(\mathcal{N}_u, \alpha_1)$
▷ *compute the unobservable SCG starting from the initial state*
3: $[\mathcal{G}, W] := \text{FDG}(\mathcal{N}, \emptyset, \mathcal{G}_{scg}, \emptyset)$
▷ *construct FDG consistent with the empty word*
4: compute the diagnosis state of α_1 in $\mathcal{G}$
5: $\mathcal{S} := \{\langle \alpha_1, \mathbf{0} \rangle\}$
▷ *next observation is obtained by exploring the output edges of states in* $\mathcal{S}$
6: let t_j be a new observation and $w = vt_j$ at τ_w
7: **if** $W \neq \emptyset$ **then**
8: **for each** $\alpha \in \mathcal{S}$ s.t. $\exists \langle \alpha, \langle \overrightarrow{\sigma t_j}, I \rangle, \alpha' \rangle \in \mathcal{G}$ $\wedge \alpha' \in W$ **do**
9: $\mathcal{G}_{scg} := \text{SCG}(\mathcal{N}_u, \alpha')$
▷ *compute the unobservable SCG starting from* α'
10: $[\mathcal{G}, W] := \text{FDG}(\mathcal{N}, \mathcal{G}, \mathcal{G}_{scg}, W)$
▷ *update the FDG by adding* $\mathcal{G}_{scg}$
11: $W := W \setminus \{\alpha'\}$ ▷ *remove explored* α' *from* W
12: **end for**
13: **end if**
14: $[D(w), \mathcal{S}] := \text{diag}(\mathcal{G}, \mathcal{S}, t_j, \tau_w, \tau_v, T_f)$
▷ τ_v *is the observation time of* v
15: **go to** 6

C. Firing Domain of A Given Firing Sequence

The *firing domain* of a firing sequence is a time interval saying the earliest and the latest instants in which all transitions belonging to the firing sequence can fire. This firing domain is used to detect if the firing sequence is consistent with the observation or not This information also appears on the edges of the FDG.

In order to compute the firing domain $[l, u]$ of a sequence σ, we apply the following steps: 1) define $|\sigma|$ variables ($|\sigma|$ is the number of transitions of σ) to represent the time when each transition is fired; 2) find when each transition is starting

to be enabled; 3) construct linear inequalities of the firing durations of the transitions (when they are enabled and when they are fired); 4) minimize and maximize the last variable, which represents the time moment when all transitions in σ have been fired, subject to the linear equations to obtain l and u.

In step 2 of Alg. 3, the sequence of states corresponding to σ is found in $\mathcal{G}_{scg}$. We associate a variable y_i to the i-th transition t_i in σ (step 3). Because the transitions in σ are sequentially fired, the time moment y_i should be in the same order (step 4). Every transition in σ is enabled either from the beginning or after a subsequence of σ. Let transition t be the last transition of σ. Assume t is enabled after the first j transitions of σ are fired, i.e., t is enabled at y_j, then the time between t is enabled to it is fired is $y_{|\sigma|} - y_j$. Because t maybe enabled multiple times until it is fired, the last time when it is enabled is used as y_j (steps 5 to 13). If a transition $t \notin \sigma$ is enabled but not fired until all transitions in σ are fired, then the firing of all transitions in σ must be earlier than t (steps 14 to 18). The firing domain of the firing sequence σ is obtained by minimizing and maximizing the last variable $y_{|\sigma|}$ subject to $cons$ (step 19).

Algorithm 3 $[g_l, g_u] := \text{domain}(\mathcal{G}_{scg}(\mathcal{N}_u, \alpha_1), \sigma, \alpha_1)$

1: $\sigma := t_1 t_2 \cdots t_{|\sigma|}$
2: get $\alpha_1 \overset{t_1}{\to} \alpha_2 \overset{t_2}{\to} \cdots \overset{t_{|\sigma|-1}}{\to} \alpha_{|\sigma|} \overset{t_{|\sigma|}}{\to} \alpha_{|\sigma|+1}$ from $\mathcal{G}_{scg}$
3: set $|\sigma|+1$ variables $y_0, y_1, \ldots, y_{|\sigma|}$
4: $cons := y_0 \le y_1 \le \cdots \le y_{|\sigma|} \wedge y_0 = 0$
5: **while** $|\sigma| > 0$ **do**
6: let σ' be the subsequence of σ by removing the last transition t
7: find max j s.t. $\sigma' := \sigma''\sigma'''$, $j = |\sigma''|$, t is enabled after σ'' and keeps to be enabled during σ'''
8: $\alpha_{j+1} := \langle \boldsymbol{m}_{j+1}, F_{j+1} \rangle$ ▷ *the state reached after all transitions in σ'' have been fired from α_1*
9: $F' := F_{j+1} \wedge (\bigwedge_{t' \in \{t_1 | t_1 \in \text{En}(\boldsymbol{m}_j), t_1 \notin \sigma'''\}} x_t \le x_{t'})$
10: $[l, u] :=$ minimize and maximize x_t s.t. F'
11: $cons := cons \wedge l \le y_{|\sigma|} - y_j \le u$
12: $\sigma = \sigma'$
13: **end while**
14: **for each** t enabled at $\alpha_{|\sigma|}$ **do**
15: find max j s.t. $\sigma = \sigma''\sigma'''$, $j = |\sigma''|$, t is enabled after σ''
16: $u :=$ maximize x_t s.t. F_{j+1}
17: $cons := cons \wedge 0 \le y_{|\sigma|} - y_j \le u$
18: **end for**
19: $[g_l, g_u] :=$ minimize and maximize $y_{|\sigma|}$ s.t. linear equalities in $cons$ are true
20: **return** $[g_l, g_u]$

Example 6: Let us consider the PN in Fig. 1, where the time durations of all transitions are $[1, 4]$, and the firing sequence $\sigma = \varepsilon_2 t_1$, which is enabled at $\alpha_1 = \langle \boldsymbol{m}_0, F_0 \rangle$ with $\boldsymbol{m}_0 = [1\ 1\ 0\ 0\ 0\ 0\ 0]^T$ and $F_0 = (1 \le x_1 \le 4) \wedge (1 \le x_2 \le 4) \wedge (1 \le x_3 \le 4)$. In order to compute the firing domain of σ, the path $\alpha_1 \overset{\varepsilon_2}{\to} \alpha_1' \overset{t_1}{\to} \alpha_5$ is found in the SCG, where $\alpha_1' = \langle [0\ 1\ 0\ 1\ 0\ 0\ 0]^T, 0 \le x_1 \le 3 \rangle$ and $\alpha_5 = \langle [0\ 0\ 0\ 1\ 1\ 0\ 0]^T, 1 \le x_4 \le 4 \rangle$. Two variables y_1 and y_2 are assigned to t_1 and ε_2, respectively, and $y_0 = 0$. From the order of transitions in σ, the order of the variables is obtained as $y_0 \le y_2 \le y_1$. Because t_1 and ε_2 are enabled from the beginning (at α_1) and are enabled until y_1 and y_2, respectively, the following inequalities are introduced $1 \le y_1 - y_0 \le 4$ and $1 \le y_2 - y_0 \le 4$. Therefore, the constrains are $0 \le y_2 \le y_1 \wedge 1 \le y_1 \le 4 \wedge 1 \le y_2 \le 4$. Minimizing and maximizing y_1 subject to the constrains, the firing domain of $\sigma = \varepsilon_2 t_1$ at α_1 is obtained as $[1, 4]$.

Algorithm 4 $[\mathcal{G}, W] = \text{FDG}(\mathcal{N}, \mathcal{G}, \mathcal{G}_{scg}, W)$

1: $\alpha_1 :=$ the initial state in $\mathcal{G}_{scg}$
2: **for each** $\alpha_u = \langle \boldsymbol{m}_u, F_u \rangle \in \mathcal{G}_{scg}$ **do**
3: **for each** $t \in \text{En}(\boldsymbol{m}_u) \cap T_o$ s.t. isFireable(α_u, t) **do**
4: $\alpha = \text{succ}(\alpha_u, t)$ ▷ *compute the successor*
5: **if** $\alpha \notin \mathcal{G}$ **then** ▷ *if α is not explored*
6: $W = W \cup \{\alpha\}$ ▷ *update W*
7: **end if**
8: $\Sigma :=$ the paths from α_1 to α_u in $\mathcal{G}_{scg}$
9: **for each** $\sigma \in \Sigma$ **do**
10: $I := \text{domain}(\mathcal{G}_{scg}, \sigma t, \alpha_1)$
11: add α and $\langle \alpha_1, \langle \vec{\sigma t}, I \rangle, \alpha \rangle$ into $\mathcal{G}$
12: **end for**
13: **end for**
14: **end for**
15: **return** $\mathcal{G}, W$

D. Updating the FDG

The computation of FDG is incremental corresponding to the observation. The update algorithm Alg. 4 accepts an unobservable SCG as a parameter. It also updates the set of unexplored states W, which is both input and output parameters of the algorithm.

Alg. 4 check every state α_u in the unobservable SCG $\mathcal{G}_{scg}$ whether there exists any observable transition that can be fired at α_u or not (step 3). If there is an observable transition $t \in T_o$ that can be fired at α_u, the successor state α of α_u is computed by firing t (step 4). If α does not belong to the FDG, i.e., α is a newly obtained state and it has not been explored, then α is added into the set of unexplored states W (step 6). After that, all paths from the initial state α_1 of $\mathcal{G}_{scg}$ to α_u are found in $\mathcal{G}_{scg}$ (step 8). Because the observable transition t is fireable at α_u, then for each path σ from α_1 to α_u, there is a corresponding path σt from α_1 to α. The firing domain I of σt is computed (step 10). The FDG is updated with the state α as a vertex and an edge $\langle \alpha_1, \langle \vec{\sigma t}, I \rangle, \alpha \rangle$ (step 11).

E. Fault Diagnosis on FDG

After updating FDG, the fault states are computed using the FDG. The diagnosis of the current observed transition starts from the state classes which are consistent with the previous observation, i.e., state classes in $\mathcal{S}$. The current

observed transition will be find in the output edges of the states in $\mathcal{S}$. The diagnosis algorithm checks the firing of fault transitions and constructs the set of consistent states corresponding to the current observed transition. Therefore, the fault diagnosis according to the observed transition computes the fault state and updates consistent states.

The fault state is initialized as normal for all fault classes (step 2). For each element $\langle \alpha, \vec{\sigma}_\alpha \rangle$ in $\mathcal{S}$, the algorithm checks whether the observed transition t is fireable at α or not by finding an edge starting at α and the firing count of t is 1 in the edge's label. If t is fireable, Alg. 5 also checks t can be fired at the observed time τ or not (step 4). If both of them are true, assuming α_1 is the target state of the edge, then the path from the intial marking to α_1 through α is consistent with the observation. Since α_1 is the state consistent with the observation, the firing count vector of the path from the initial marking to α_1 is computed as $\vec{\sigma}_{\alpha_1}$ and $\langle \alpha_1, \vec{\sigma}_{\alpha_1} \rangle$ is inserted into $\mathcal{S}_f$, which is the set of consistent marking with the current observation and it is initialized as an empty set at the beginning of Alg. 5 (steps 5 and 6). The fault state $D(w)$ is updated using the firing vector of the path from the initial marking to α_1 and the output edges of α_1 (step 7). The following steps are performed to update the fault state: 1) First, the fault occurence in the path from the initial state to α_1 and the output edges of α_1 is computed. Considering $T_f^i = \{t_f\}$, if the firing count of t_f in $\vec{\sigma}_{\alpha_1}$ is not zero, then it has been fired and the diagnosis state of T_f^i corresponding to α_1 is fault. If t_f is not fired in the path, then check the firing counts of t_f in the output edges of α_1. If all the firing counts are greater than zero, then the diagnosis state of T_f^i is fault; if they are zero, then the diagnosis state is normal; otherwise, the diagnosis state is uncertain. 2) Second, the diagnosis state computed in the first step is applied to update $D(w)$. If the fault state of T_f^i in $D(w)$ is normal and the fault state in the first step is normal too, then the updated fault state of T_f^i is normal, because t_f has not been fired in any firing sequence which is consistent with the observation. If both of the fault states are fault, then the updated fault state in $D(w)$ is fault, because t_f has been fired in all firing sequences which are consistent with the observation. Otherwise, the updated fault state is uncertain.

F. Discussion

A time PN has a bounded number of state classes if it is bounded and bounds of time durations are rational for all transitions [10]. The boundedness problem of number of state classes of time PNs is undecidable [7]. The FDG is completely constructed when the set of unexplored states (W) is empty, i.e., all states have been explored, and it is the time when all state classes have been generated. In each execution of Alg. 4, the algorithm updates the FDG using an unobservable SCG that it finds paths from the initial state class of the unobservable SCG to every state at which an observable can be fired from. In the worst case, if at every state, an observable transition can be fired, then the number of states in the FDG is the same as the SCG.

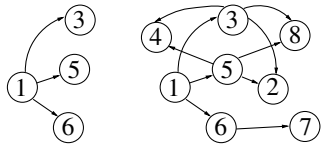
Algorithm 5 $[D(w), \mathcal{S}] = \text{diag}(\mathcal{G}, \mathcal{S}, t, \tau, \tau_v, T_f)$

1: $\mathcal{S}_f := \emptyset$
2: $D(w) := \{\langle i, \text{N} \rangle | i = 1, \ldots, r\}$
   ▷ $1, \ldots, r$ *are the number of fault classes*
3: **for each** $\langle \alpha, \vec{\sigma}_\alpha \rangle \in \mathcal{S}$ **do**
4: **for each** $\langle \alpha, \langle \overrightarrow{\sigma_u t}, I \rangle, \alpha_1 \rangle \in \mathcal{G}$ s.t. $\tau - \tau_v \in I$ **do**
   ▷ τ *and* τ_v *are the observed time of* t *and the previous observed transition*
5: $\vec{\sigma}_{\alpha_1} := \vec{\sigma}_\alpha + \overrightarrow{\sigma_u t}$
6: $\mathcal{S}_f := \mathcal{S}_f \cup \{\langle \alpha_1, \vec{\sigma}_{\alpha_1} \rangle\}$
7: update $D(w)$ using $\vec{\sigma}_{\alpha_1}$ and the output edges of α_1
8: **end for**
9: **end for**
10: $\mathcal{S} := \mathcal{S}_f$
11: **return** $[D(w), \mathcal{S}]$

The approach in [9] computes SCG, which is a compact representation of the reachability graph of a TPN. In this paper, we optimize the SCG by removing unnecessary states for fault diagnosis. In [5], an ILP based approach is applied. The set of constraints may become very large, because after each observation some constraints are inserted. We use an enumeration based approach and ILP is not used in diagnosis. Linear programming problem is applied to compute the firing domain of a firing sequence. In [11], a net unfolding approach is applied to the diagnosis of systems modeled by TPN. Since the approach is based on net unfolding, they focus on 1-bounded PNs.

V. Example

Consider the PN model in Fig. 1 with $\boldsymbol{m}_0 = [1\ 1\ 0\ 0\ 0\ 0\ 0]^T$. The time intervals of all transitions are $[1, 4]$. The unobservable subnet $\mathcal{N}_u$ contains $T_u = \{\varepsilon_2, \varepsilon_3, \varepsilon_4, \varepsilon_6\}$. The fault transition is ε_4 and the fault class $T_f^1 = \{\varepsilon_4\}$. The initial state class is $\alpha_1 = \langle \boldsymbol{m}_0, 1 \leq x_1 \leq 4 \wedge 1 \leq x_2 \leq 4 \wedge 1 \leq x_3 \leq 4 \rangle$. Let us assume t_1 is observed at time 3.



(a) $w = \epsilon$ (b) $w = t_1$

Fig. 3. FDG of the model in Fig. 1

In the diagnosis corresponding to $w = \epsilon$, the unobservable SCG of $\mathcal{N}_u$ with initial state as α_1 is computed by applying Alg. 1. The FDG in Fig. 3(a) is constructed based on the unobservable SCG (Alg. 4). The vertices of the FDG and labels of edges are shown in Tab. I and Tab. II, respectively. The last column in Tab. II are firing sequences of corresponding edges and these information are not included to the labels. We compute the whole SCG of the full net, which has 20 state classes, and they are reduced to 10 states

TABLE I

STATE CLASSES (VERTICES) IN THE FDG

state	marking	firing domain
α_1	$[1\ 1\ 0\ 0\ 0\ 0\ 0]^T$	$1 \le x_1 \le 4 \wedge 1 \le x_2 \le 4 \wedge 1 \le x_3 \le 4$
α_2	$[1\ 1\ 0\ 0\ 0\ 0\ 0]^T$	$0 \le x_1 \le 4 \wedge 1 \le x_2 \le 4 \wedge 1 \le x_3 \le 4$
α_3	$[1\ 0\ 0\ 0\ 1\ 0\ 0]^T$	$0 \le x_2 \le 3 \wedge 0 \le x_3 \le 3$
α_4	$[0\ 0\ 0\ 0\ 1\ 1\ 0]^T$	$0 \le x_7 \le 2$
α_5	$[0\ 0\ 0\ 1\ 1\ 0\ 0]^T$	$1 \le x_4 \le 4$
α_6	$[0\ 0\ 1\ 0\ 1\ 0\ 0]^T$	$1 \le x_5 \le 4$
α_7	$[0\ 0\ 0\ 0\ 0\ 1\ 1]^T$	$1 \le x_6 \le 4 \wedge 1 \le x_7 \le 4$
α_8	$[1\ 0\ 0\ 0\ 0\ 0\ 1]^T$	$1 \le x_2 \le 4 \wedge 1 \le x_3 \le 4 \wedge 0 \le x_6 \le 3$

TABLE II

LABELS OF EDGES

edge	firing count vector	firing domain	firing sequences
$\alpha_1 \to \alpha_3$	$[1\ 0\ 0\ 0\ 0\ 0\ 0]$	$[1, 4]$	t_1
$\alpha_1 \to \alpha_5$	$[1\ 1\ 0\ 0\ 0\ 0\ 0]$	$[1, 4]$	$\varepsilon_2 t_1$
$\alpha_1 \to \alpha_6$	$[1\ 0\ 1\ 0\ 0\ 0\ 0]$	$[1, 4]$	$\varepsilon_3 t_1$
$\alpha_3 \to \alpha_2$	$[0\ 1\ 0\ 1\ 0\ 1\ 1]$	$[2, 11]$	$\varepsilon_2\varepsilon_4\varepsilon_6 t_7$
$\alpha_3 \to \alpha_7$	$[0\ 0\ 1\ 0\ 1\ 0\ 0]$	$[1, 7]$	$\varepsilon_3 t_5$
$\alpha_3 \to \alpha_8$	$[0\ 1\ 0\ 1\ 0\ 0\ 1]$	$[2, 11]$	$\varepsilon_2\varepsilon_4 t_7$
$\alpha_3 \to \alpha_4$	$[1\ 1\ 0\ 1\ 0\ 1\ 0]$	$[3, 15]$	$\varepsilon_2\varepsilon_4\varepsilon_6 t_1$
$\alpha_5 \to \alpha_2$	$[0\ 0\ 0\ 1\ 0\ 1\ 1]$	$[2, 8]$	$\varepsilon_4\varepsilon_6 t_7$
$\alpha_5 \to \alpha_8$	$[0\ 0\ 0\ 1\ 0\ 0\ 1]$	$[2, 8]$	$\varepsilon_4 t_7$
$\alpha_5 \to \alpha_4$	$[1\ 0\ 0\ 1\ 0\ 1\ 0]$	$[3, 12]$	$\varepsilon_4\varepsilon_6 t_1$
$\alpha_6 \to \alpha_7$	$[0\ 0\ 0\ 0\ 1\ 0\ 0]$	$[1, 4]$	t_5

in the FDG. For the considered observation in this section, only 8 of them are used. Three new obtained states α_3, α_5 and α_6 are added in to W, because they are not explored (step 6 in Alg. 4). At every time moment when a transition is observed, the explored states are removed from W and newly generated states, which have not been explored, are inserted into W. Take the edge from state α_1 to α_5 as example. Its firing count vector is $\overrightarrow{\varepsilon_2 t_1}$, since the firing sequence is $\varepsilon_2 t_1$. Applying Alg. 3, its firing domain is computed as $[1, 4]$. The firing count vectors of three output edges are considered to compute the fault state correpsonding to $w = \epsilon$. Because the firing count of the fault transition ε_4 is 0 in every edge, the fault state is normal of T_f^1 (Alg. 5). The set $\mathcal{S}$ is initialized with the initial state (step 5 in Alg. 2). After the diagnosis, there are $W = \{\alpha_3, \alpha_5, \alpha_6\}$ and $\mathcal{S} = \{\langle \alpha_1, \vec{0} \rangle\}$.

The observed transition is t_1 at 3. Alg. 2 checks at which state in $\mathcal{S}$ the observed transition t_1 can fire. The state α_1 in $\mathcal{S}$ has three output edges and t_1 is fired in each of them. The edges are from α_1 to α_3, α_5 and α_6, respectively. The unobservable SCGs of them are computed whose initial states are α_3, α_5 and α_6, respectively, and the FDG (Fig. 3(b)) is updated based on the SCGs (steps 9 to 11 in Alg. 2). In the update of the FDG, because α_3 is explored, it is removed from the set of unexplored states W. Four states α_2, α_7, α_8 and α_9 are newly obtained and have not been explored, then W is updated to $\{\alpha_2, \alpha_7, \alpha_8, \alpha_9\}$ (step 6 in Alg. 4). After the update of the FDG, the diagnosis algorithm Alg. 5 computes the fault states. In order to explain how does the diagnosis algorithm works, we take the edge from α_1 to α_3 as an example. Because the firing domain of the edge is $[1, 4]$ and t_1 is observed at 3, α_3 is a consistent state of $w = t_1$ (step 4 in Alg. 5). By adding the path vector of α_1, which is $\vec{0}$, and the firing count vector of the edge from α_1 to α_3, the firing count vector of the path from the initial state α_1 to α_3 is computed, which is $\vec{t_1}$ and the fault transition ε_4 is not fired in this path (step 5 in Alg. 5). The state α_3 and the vector computed in previous step are added into $\mathcal{S}_f$, which is the set of consistent states of $w = t_1$ (step 6 in Alg. 5). The output edges of α_3 provides the diagnosis information of the unobservable SCG of α_3 (step 7 in Alg. 5). The information is that the fault transition ε_4 is not fired in the edge from α_3 to α_7 and it is fired in the remaining output edges of α_3. This information and the firing count vector of the path from α_1 to α_3 are used to update the fault state corresponding to $w = t_1$ and the fault state of T_f^1 is U. After other two output edges of α_1 have been considered in the same way, the fault state corresponding to $w = t_1$ is uncertain of T_f^1. After the diagnosis, the set of unexplored states $W = \{\alpha_2, \alpha_7, \alpha_8, \alpha_9\}$ and the set $\mathcal{S}$ is updated to $\mathcal{S} = \{\langle \alpha_3, \vec{t_1} \rangle, \langle \alpha_5, \overrightarrow{\varepsilon_2 t_1} \rangle, \langle \alpha_6, \overrightarrow{\varepsilon_3 t_1} \rangle\}$.

VI. CONCLUSIONS

In this paper, we have investigated the problem of fault diagnosis on time PN and we propose the use of the Fault Diagnosis Graph (FDG). The fault diagnosis on time PN benefits from FDG. First, states and firing of transitions are not put in FDG if they are not used in the fault diagnosis. The number of states in the FDG is smaller than the one of the state class graph. Second, paths are stored in FDG and they are reusable when repetitive behaviors appear. The repetitive search operations of same path are avoided. In future work, we plan to implement the algorithms in Matlab.

REFERENCES

[1] D. Corona, A. Giua, and C. Seatzu, "Marking estimation of Petri nets with silent transitions," *IEEE Transaction on Automatic Control*, vol. 52, no. 9, pp. 1695–1699, 2007.

[2] M. Dotoli, M. Fanti, A. Mangini, and W. Ukovich, "On-line fault detection in discrete event systems by Petri nets and integer linear programming," *Automatica*, vol. 45, no. 11, pp. 2665–2672, 2009.

[3] X. Wang, C. Mahulea, J. Jùlvez, and M. Silva, "On state estimation of timed choice-free Petri nets," in *Proceedings of the 18th IFAC World Congress*, 2011.

[4] 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, 2010.

[5] M. Fanti, A. Mangini, and W. Ukovich, "Fault detection by labeled Petri nets and time constraints," in *3rd International Workshop on Dependable Control of Discrete Systems (DCDS)*, 2011, pp. 168–173.

[6] F. Basile, P. Chiacchio, and G. De Tommasi, "Improving on-line fault diagnosis for discrete event systems using time," in *IEEE International Conference on Automation Science and Engineering*, 2007, pp. 26–32.

[7] B. Berthomieu and M. Menasche, "An enumerative approach for analyzing time Petri nets," in *IFIP Congress Series*, vol. 9, 1983, pp. 41–46.

[8] P. Merlin and D. Farber, "Recoverability of communication protocols–implications of a theoretical study," *IEEE Transactions on Communications*, vol. 24, no. 9, pp. 1036 – 1043, 1976.

[9] R. Hadjidj and H. Boucheneb, "Efficient Reachability Analysis for Time Petri Nets," *IEEE Transactions on Computers*, vol. 60, no. 8, pp. 1085 –1099, 2011.

[10] B. Berthomieu and M. Diaz, "Modeling and verification of time dependent systems using time Petri nets," *IEEE Transactions on Software Engineering*, vol. 17, no. 3, pp. 259 –273, 1991.

[11] G. Jiroveanu, R. Boel, and B. De Schutter, "Fault diagnosis for time Petri nets," in *Proceedings of the 8th International Workshop on Discrete Event Systems (WODES'06)*, 2006, pp. 313–318.