2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Diagnosis of Time-Sensitive Failures in Timed Discrete-Event Systems with Metric Interval Temporal Logics

Weijie Dong, Shaoyuan Li and Xiang Yin

Abstract— **In this paper, we address the problem of failure diagnosis in timed discrete-event systems modeled by timed automata. While existing works on this topic typically focus on failures modeled as particular events, many complex applications, especially time-critical systems, require the ability to identify *time-sensitive failures* associated with real-time information rather than just the occurrence of events at any time. To address this challenge, we propose the use of metric interval temporal logic (MITL) with continuous semantics on Boolean signals to formally describe time-sensitive failures. We introduce a novel concept called *time-sensitive diagnosability* (TS-diagnosability) to characterize whether or not any violation of the MITL task (i.e., failure) can be determined within a finite time elapsing. Furthermore, we provide a necessary and sufficient condition for verifying TS-diagnosability. Our results offer a more general framework for failure diagnosis of timed discrete-event systems.**

I. Introduction

Engineering cyber-physical systems (CPS), such as manufacturing systems and intelligent transportation systems, are critical infrastructures of modern society. These systems, which rely on complex connections among millions of components and modules, require high-levels of safety. However, their intricate operation logic make CPS vulnerable to malfunction. As a result, *failure diagnosis and detection* are crucial but challenging tasks to ensure the safety and reliability of CPS. In this paper, we investigate failure diagnosis problem in the context of discrete-event systems (DES); see, e.g., some recent works [5], [7], [13]–[16], [19]–[22] and survey paper [11].

Real-world engineering systems often have strict time constraints that must be followed for proper behavior execution. This *real-time* information is crucial for effective fault diagnosis in these systems. The fault diagnosis problem for timed systems has been extensively studied, as seen in works such as [3], [4], [18], [23]. In [18], the concept of diagnosability was introduced for timed systems modeled by timed automata [1], and a diagnoser was developed as an online algorithm that tracks all possible states and zones. In [4], the diagnoser was realized as a deterministic timed automaton. In [23], the authors explicitly modeled the tick of a global clock by an event and investigated fault diagnosis for automata containing the tick event. In [3], diagnosability was analyzed for labeled time Petri net systems.

The aforementioned works on diagnosis of DES have only considered faults modeled by a single event or state. However, in practice, many engineering systems have real-time and temporal constraints that are necessary for achieving their tasks. Such systems are referred to as *time-critical systems* because any violation of the desired real-time constraint can lead to failures. Unfortunately, these complex time-sensitive failures cannot be captured using a single fault event in existing frameworks.

To address the above mentioned limitations, we propose a novel approach that utilizes metric interval temporal logic (MITL) [2] to formally describe time-sensitive failures in timed DES modeled by a timed automaton. In this approach, any violation of the given MITL specification is considered a failure. We introduce a new concept called *time-sensitive diagnosability* (TS-diagnosability) to determine whether any time-sensitive failure can be diagnosed within a finite time. Furthermore, we provide a necessary and sufficient condition for verifying TS-diagnosability by leveraging the effective translation from MITL formulae to timed transducers. To the best of our knowledge, MITL has not been used to describe failures in the context of timed DES before.

It is worth noting that, there are several methods in the literature for representing system failures. For instance, supervision patterns were utilized in [9], [10] to model complex faulty behaviors. In [6], authors adopted Linear Temporal Logic (LTL) formulae to define system failures and studied the diagnosability verification problem. However, these approaches were limited to untimed systems, and they cannot capture the concept of *time-sensitive failures* that arise when behaviors of timed systems violate strict time constraints. Therefore, our new approach provides a more expressive framework for the diagnosis of time-critical systems, which has not been explored in the literature on DES so far.

II. Preliminaries

A. System Model

A *time interval* Δ is a convex subset of $\mathbb{R}_+$ in the following forms: $[0,a], [0,a), (0,a], (0,a), (0,\infty)$ and $[0,\infty)$, where $a \in \mathbb{R}_+ \setminus \{0\}$. In particular, we define the *singular time interval* as $\Delta = [0,0]$. The length of time interval Δ is exactly the right-end point, denoted by $|\Delta|$, e.g., $|[0,a)| = a$. Two time intervals Δ_1 and Δ_2 are *adjacent* if either Δ_1 is right open and Δ_2 is left closed, or Δ_1 is right closed and Δ_2 is left open. A time interval sequence $\Delta_0\Delta_1\Delta_2\cdots$ is *well-defined* if for any $i \geq 0$, the time intervals Δ_i and

This work was supported by the National Natural Science Foundation of China (62061136004, 62173226, 61803259).

W. Dong, S. Li and X. Yin are with Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: {wjd_dollar,syli,yinxiang}@sjtu.edu.cn.

979-8-3503-0123-6/23/$31.00 ©2023 IEEE

Δ_{i+1} are adjacent. In this paper, we only consider well-defined time interval sequences. Let Σ be a finite alphabet. We define a signal over Σ by $\mathsf{s} = (p_0, \Delta_0)(p_1, \Delta_1)\cdots$, where $\Delta_0\Delta_1\cdots$ is a time interval sequence and p_i is a subset of Σ, for any $i \geq 0$. We denote by $\mathsf{Sig}(\Sigma)$ the set of all signals over Σ. The time length of signal s is denoted by $\mathsf{time}(\mathsf{s}) = \sum_{i\in\{0,1,\cdots\}} |\Delta_i|$. A signal is infinite if its time length is infinite; otherwise it is finite. For finite set Σ, we denote by $|\Sigma|$ its cardinality. A sequence $s = q_1 q_2 \cdots q_n(\cdots)$ is a finite (or infinite) sequence over Σ if $q_i \in \Sigma$ for all $i \geq 0$. We denote by Σ^* and Σ^ω the set of all finite and infinite sequence over Σ, respectively.

Let $\mathcal{X}$ denotes the set of clock variables whose codomain is $\mathbb{R}_+$. Clock valuation v is a function assigning each clock variable a non-negative real number, i.e., $v : \mathcal{X} \to \mathbb{R}_+$. We denote by $\mathcal{V}_\mathcal{X}$ the set of all clock valuations over clock set $\mathcal{X}$, and by $\mathbf{0}_\mathcal{X}$ the valuation that assigns 0 to every clock. Given a clock valuation $v \in \mathcal{V}_\mathcal{X}$ and a real number $t \in \mathbb{R}_+$, we define the valuation $v+t$ by $(v+t)(c) = v(c)+t$, for any $c \in \mathcal{X}$. For a subset of clock variables $\mathcal{Y} \subseteq \mathcal{X}$, we denote by $v_{[\mathcal{Y}\leftarrow 0]}$ the valuation that resets clock variables in $\mathcal{Y}$ to 0 and reserves values of other clocks. Given a clock valuation $v : \mathcal{X} \to \mathbb{R}_+$ and a subset $\mathcal{X}' \subseteq \mathcal{X}$, we say $v_{\mathcal{X}'} : \mathcal{X}' \to \mathbb{R}_+$ is a *reduced valuation* if $\forall c \in \mathcal{X}' : v_{\mathcal{X}'}(c) = v(c)$.

An *atomic clock constraint* of a clock variable $c \in \mathcal{X}$ is of the form $c \sim a$, where $a \in \mathbb{N}_+$ is a constant and $\sim \in \{<, \leq, =, \geq, >\}$. A valuation $v \in \mathcal{V}_\mathcal{X}$ satisfies an atomic clock constraint $c \sim a$ whenever $v(c) \sim a$. A *clock constraint* g is a Boolean combination of atomic clock constraints. We denote by $C(\mathcal{X})$ the set of all clock constraints over clock set $\mathcal{X}$. Given a clock constraint $g \in C(\mathcal{X})$, a valuation $v \in \mathcal{V}_\mathcal{X}$ satisfies g is denoted by $v \models g$.

In this paper, we consider a timed system modeled by a variant of timed automaton, called interval automaton (IA) [1], which is a seven-tuple

$$G = (Q, Q_0, \mathcal{X}, \mathsf{inv}, E, \mathcal{AP}, L), \tag{1}$$

where Q is a finite set of discrete states, $Q_0 \subseteq Q$ is the set of initial discrete states, $\mathcal{X}$ is a finite set of clocks, $\mathsf{inv} : Q \to C(\mathcal{X})$ is *invariant function* which assigns each state $q \in Q$ a clock constraint $\mathsf{inv}(q)$ specifying the length of time the system can stay at q, $E \subseteq Q \times C(\mathcal{X}) \times 2^\mathcal{X} \times Q$ is the transition relation such that transition $(q, g, \mathcal{Y}, q') \in E$ indicates a transition from q to q' satisfying the *guard* g and resetting all clocks in set $\mathcal{Y} \subseteq \mathcal{X}$ to zero after this transition, $\mathcal{AP}$ is a finite set of atomic propositions and $L : Q \to 2^{\mathcal{AP}}$ is a labeling function assigning a set of atomic propositions to each discrete state. To formalize the semantics of an IA G, we define the time state (or simply state) of G as a pair $s = (q, v)$ where $q \in Q$ and $v \in \mathcal{V}_\mathcal{X}$. We denote by $S(G) \subseteq Q \times \mathcal{V}_\mathcal{X}$ the set of all states in G. The set of initial states are denoted by $S_0(G) = \{(q_0, \mathbf{0}_\mathcal{X}) : q_0 \in Q_0\}$. There are two types of state transitions in an IA G, discrete transition and time transition, which are defined by: for any states $(q, v), (q', v') \in S(G)$ and a time interval Δ,

- *time transition*: there is a transition $(q, v) \xrightarrow{\Delta} (q, v + |\Delta|)$ whenever for any $t \in \Delta, v + t \models \mathsf{inv}(q)$;
- *discrete transition*: there is a discrete transition $(q, v) \xrightarrow{\sigma} (q', v')$ whenever there exists a transition $(q, g, \mathcal{Y}, q') \in E$ such that $v \models g$ and $v' = v_{[\mathcal{Y}\leftarrow 0]}$.

In particular, we denote by $(q, v) \xrightarrow{\varepsilon} (q, v)$ the empty transition. For simplicity, we consider *mixed transition* $(q, v) \xrightarrow{(\Delta,\delta)} (q', v')$ which represents that there is another state (q'', v'') such that $(q, v) \xrightarrow{\Delta} (q'', v'')$ is a time transition and $(q'', v'') \xrightarrow{\delta} (q', v')$ is a discrete transition or empty transition, where $\delta \in \{\sigma, \varepsilon\}$.

The executions of an IA is modeled by *runs*. A run of G starting from state $s = (q, v) \in S(G)$ is a sequence

$$\pi = \xrightarrow[v_0]{\delta_0} (q_0, \Delta_0) \xrightarrow[v_1]{\delta_1} (q_1, \Delta_1) \xrightarrow[v_2]{\delta_2} (q_2, \Delta_2) \cdots$$

satisfying following conditions: (i) *initial condition*: $q_0 = q, v_0 = v$; (ii) *time condition*: $\Delta = \Delta_0\Delta_1\cdots$ is a time interval sequence; (iii) *transition condition*: for any $i \geq 0$, $\delta_i \in \{\sigma, \varepsilon\}$ and the transition $(q_i, v_i) \xrightarrow{(|\Delta_i|, \delta_{i+1})} (q_{i+1}, v_{i+1})$ holds. Given a run π, we define $\mathsf{time}(\pi) = \sum_{i\in\{0,1,\cdots\}} |\Delta_i|$ as the total time elapsing of π. Run π is said to be infinite if $\mathsf{time}(\pi) = \infty$; otherwise π is finite. Given an IA G, we assume each run $\pi \in \mathsf{Run}^*(G)$ obeys *finite-variability* [2], that is, during finite time elapsing, there are finite discrete transitions. A state $(q, v) \in S(G)$ is said to be a *timelock* [17] if there are no infinite runs starting from it and G is *timelock-free* if all of its reachable states are not timelock. We assume that the IA is timelock-free in this paper.

For a finite run $\pi = \xrightarrow[v_0]{\delta_0} (q_0, \Delta_0) \xrightarrow[v_1]{\delta_1} \cdots \xrightarrow[v_n]{\delta_n} (q_n, \Delta_n)$, we denote by $\mathsf{last}(\pi) = (q_n, v_n + |\Delta_n|)$ the last state in π, and by $\mathsf{last}_d(\pi) = q_n$ its last discrete sate. Let $S_1 \subseteq S(G)$ be a state set in G. The set of finite runs and infinite runs starting from states in S_1 are denoted by $\mathsf{Run}^*(G, S_1)$ and $\mathsf{Run}^\omega(G, S_1)$, respectively; we denote the set of all runs starting from S_1 by $\mathsf{Run}(G, S_1) = \mathsf{Run}^*(G, S_1) \cup \mathsf{Run}^\omega(G, S_1)$. For simplicity, we write $\mathsf{Run}^*(G)$ and $\mathsf{Run}^\omega(G)$ to denote $\mathsf{Run}^*(G, S_0(G))$ and $\mathsf{Run}^\omega(G, S_0(G))$, respectively. The set of all runs generated by G is denoted by $\mathsf{Run}(G) = \mathsf{Run}^*(G) \cup \mathsf{Run}^\omega(G)$. The concatenation of two runs $\pi, \pi' \in \mathsf{Run}(G, S(G))$ is denoted by $\pi\pi' \in \mathsf{Run}(G, S(G))$. Given a set of runs $\Pi \subseteq \mathsf{Run}(G)$, we define the prefix-closure of Π by $\mathsf{Pre}(\Pi) = \{\pi' \in \mathsf{Run}^*(G) : \exists \pi \in \Pi, \exists \pi'' \in \mathsf{Run}(G, S(G))$ s.t. $\pi'\pi'' = \pi\}$. For any run $\pi \in \mathsf{Run}(G)$, we write $\mathsf{Pre}(\{\pi\})$ as $\mathsf{Pre}(\pi)$ for the sake of simplicity.

Given a run π, we denote by $s_\pi = (q_0, \Delta_0)(q_1, \Delta_1)\cdots$ the timed state sequence of π and denote by $\mathsf{Path}(G) = \{s_\pi : \pi \in \mathsf{Run}(G)\}$ all timed state sequences of G. Using labeling function, we obtain timed trace $\mathsf{trace}(\pi) \in \mathsf{Sig}(\mathcal{AP})$ of run π by replacing each discrete state q_i in s_π with $L(q_i)$, for $i \geq 0$, i.e., $\mathsf{trace}(\pi) = (L(q_0), \Delta_0)(L(q_1), \Delta_1)(L(q_2), \Delta_2)\cdots$. We denote by $\mathcal{L}(G)$ all timed traces generated by system G. For an IA G with clock set $\mathcal{X}$, we denote by $c_{\mathcal{X}'}(G)$ the largest integer a such that $c \sim a \in C(\mathcal{X}')$ is a subformula of some clock constraint in G, where $\mathcal{X}' \subseteq \mathcal{X}$ and $\sim \in \{\leq, <, \geq, >, =\}$. Then, we define the untimed state sequence of a run $\pi \in \mathsf{Run}(G)$ by a mapping $\mathsf{unt} : \mathsf{Run}^*(G) \to Q^*$ such that the mapping only keeps discrete states the run visits.

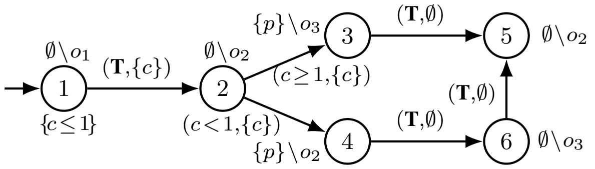


Fig. 1: Interval automaton G. For each guard, $\mathbf{T}$ is the abbreviation of `true`. The conjunction of elements in the set next to discrete state is invariant of the discrete state, and if the invariant is `true`, we omit it.

Formally, given $\pi \in \mathsf{Run}^*(G)$, unt is defined recursively by: (i) for $\pi = \xrightarrow[v]{\delta} (q, \Delta)$, we have $\mathsf{unt}(\pi) = q$; (ii) for $\pi = \pi' \xrightarrow[v]{\delta} (q, \Delta)$, we have $\mathsf{unt}(\pi) = \mathsf{unt}(\pi')$ if $\delta = \varepsilon$, and $\mathsf{unt}(\pi) = \mathsf{unt}(\pi')q$, if $\delta = \sigma$.

In the context of failure diagnosis, the observation is not perfect. Given an IA $G = (Q, Q_0, \mathcal{X}, \mathsf{inv}, E, \mathcal{AP}, L)$, we define by $\mathcal{O}$ the set of output symbols. Furthermore, we assume the external observer can measure the global time. Then we define the output function as $H : Q \to \mathcal{O}$. Given a run $\pi = \xrightarrow[v_0]{\delta_0} (q_0, \Delta_0) \xrightarrow[v_1]{\delta_1} (q_1, \Delta_1) \cdots$, the observation of π is denoted by $\rho_\pi = (H(q_0), \Delta_0)(H(q_1), \Delta_1)(H(q_2), \Delta_2) \cdots$.

Example 1: Let us consider IA G, depicted in Figure 1, where we have $\mathcal{AP} = \{p\}$ and $\mathcal{O} = \{o_1, o_2, o_3\}$. The labeling function L and output function H are specified by the label next to each state, where the LHS of "\" represents the set of atomic propositions assigned to this state modeling the internal behaviors of the system, and the RHS of "\" represents output label generated by this state. For example, $\{p\}\backslash o_3$ above state 3 means that the atomic proposition p is true at state 3 and we observe label o_3 at this point. Then, we can obtain a run $\pi = \xrightarrow[\mathbf{0}_{\mathcal{X}}]{\varepsilon} (1, [0, 1]) \xrightarrow[v_1]{\sigma} (2, (0, 1])) \xrightarrow[v_2]{\varepsilon} (2, (0, 0.5])) \xrightarrow[v_3]{\sigma} (3, (0, 0.5])$. The time elapsing of π is $\mathsf{time}(\pi) = 3$. Then, we can obtain its timed state sequence $s_\pi = (1, [0, 1])(2, (0, 1])(2, (0, 0.5])(3, (0, 0.5])$ and the untimed state sequence of π is $\mathsf{unt}(\pi) = 123$. The timed trace and observation of π are $\mathsf{trace}(\pi) = (\emptyset, [0, 1])(\emptyset, (0, 1])(\emptyset, (0, 0.5])(p, (0, 0.5])$ and $\rho_\pi = (o_1, [0, 1])(o_2, (0, 1.5])(o_3, (0, 0.5])$, respectively.

B. Region Automata

We briefly review the region automata [1]. We first define the clock regions of IA G. A clock region is a class of equivalent valuations and we denote by $\mathcal{R}(G)$ the set of all clock regions of G. For a valuation $v \in \mathcal{V}_{\mathcal{X}}$, the unique clock region corresponding to v in $\mathcal{R}(G)$ is denoted by $[v]_G$. Given two clock region $r, r' \in \mathcal{R}(G)$, r' is the successor region of r if r' can be obtained by time elapsing. For a clock region $r \in \mathcal{R}(G)$ and a clock subset $\mathcal{X}' \subseteq \mathcal{X}$, we define *reduced clock region* by $r(\mathcal{X}') = \{v_{\mathcal{X}'} : v \in r\}$. Specifically, the region automaton of G is a 4-tuple

$$RA(G) = (Q^R, Q_0^R, \Sigma^R, E^R),$$

where $Q^R = Q \times \mathcal{R}(G)$ is a finite set of states, $Q_0^R = \{(q_0, [\mathbf{0}_{\mathcal{X}}]_G) : q_0 \in Q_0\}$ is the set of initial states, $\Sigma^R = \{\tau, \sigma\}$ is a set of events and $E^R : Q^R \times \Sigma^R \to 2^{Q^R}$ is the transition function, which is defined by: for $(q, r), (q', r') \in Q^R$ and $\lambda \in \Sigma^R$, we have transition $(q', r') \in E^R((q, r), \lambda)$ if (i) $\lambda = \sigma$ and there is a discrete transition $(q, v) \xrightarrow{\sigma} (q', v')$ for $v \in r$ and $v' \in r'$; or (ii) $\lambda = \tau$ and there is a time transition $(q, v) \xrightarrow{\Delta} (q', v')$ for $v \in r$ and $v' \in r'$, where r' is a time successor of r.

A run in $RA(G)$ is a finite (or infinite) sequence $\pi = q_0^R \xrightarrow{\lambda_0} q_1^R \xrightarrow{\lambda_1} \cdots q_n^R(\cdots)$, where $q_i^R \in Q^R, \lambda_i \in \Sigma^R$ and $q_{i+1}^R \in E^R(q_i^R, \lambda_i)$, for $i = 0, 1, \cdots, n(\cdots)$. We denote by $\mathsf{Run}^R(G)$ all runs generated by $RA(G)$. Given a finite run $\pi = q_0^R \xrightarrow{\lambda_0} \cdots q_n^R$, we denote the last state of π by $\mathsf{last}_R(\pi) = q_n^R$ and we say π is a cycle if $q_n^R = q_0^R$. The region automaton abstracts time transitions of the original system and preserves the discrete transitions. Specifically, we define a mapping $\mathsf{unt}^R(\pi^R) : \mathsf{Run}^R(G) \to Q^*$ for run $\pi^R \in \mathsf{Run}^R(G)$ such that the mapping only reserves discrete state component in states that are visited by π^R. Formally, unt^R is defined recursively by: (i) for $\pi^R = (q, r) \in \mathsf{Run}^R(G)$, we have $\mathsf{unt}^R(\pi^R) = q$; (ii) for $\pi^R = \pi' \xrightarrow{\lambda} (q, r) \in \mathsf{Run}^R(G)$ we have $\mathsf{unt}^R(\pi^R) = \mathsf{unt}^R(\pi')q$ if $\lambda = \sigma$, and $\mathsf{unt}^R(\pi^R) = \mathsf{unt}^R(\pi')$ if $\lambda = \tau$. It is known that region automaton $RA(G)$ generates the same untimed runs with the original IA G [1]. Specifically, there exists a run $\pi \in \mathsf{Run}^*(G)$ and $\mathsf{last}(\pi) = (q, v)$ iff there is a run $\pi^R \in \mathsf{Run}^R(G)$ such that $\mathsf{unt}^R(\pi^R) = \mathsf{unt}(\pi)$ and $\mathsf{last}_R(\pi^R) = (q, [v]_G)$.

C. Metric Interval Linear Temporal Logic

Let $\mathcal{AP}$ be a finite set of atomic propositions. A Metric Interval Linear Temporal Logic (MITL) formula φ is constructed based on the following syntax [2]:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathcal{U}_I \varphi_2,$$

where $\mathcal{U}_I$ denotes the "until" temporal operator. Here $I \subseteq \mathbb{Q}_{\geq 0}$ is an interval of non-negative rational numbers with integer end-points, and we restrict that I is not a singleton. Based on the fundamental syntax, we can derive other Boolean operators, e.g., $\varphi_1 \to \varphi_2 = \neg\varphi_1 \vee \varphi_2$. Similarly, we can induce temporal operators $\Diamond_I$ "eventually" by $\Diamond_I \varphi = \top \mathcal{U}_I \varphi$ and $\Box_I$ "always" by $\Box_I \varphi = \neg\Diamond_I \neg\varphi$.

MITL formulae are evaluated on infinite signals over atomic proposition set. The reader is referred to [8] for more details on the continuous-time semantics of MITL. We denote by $\mathsf{s} \models \varphi$ if a signal $\mathsf{s} \in \mathsf{Sig}(\Sigma)$ satisfies formula φ. We denote by $\mathsf{signal}(\varphi)$ the set of all signals satisfying the formula φ. We say that a run $\pi \in \mathsf{Run}^\omega(G)$ satisfies specification φ if its timed trace satisfies φ, i.e., $\mathsf{trace}(\pi) \models \varphi$. For simplicity, with a slight abuse of notations, we write $\pi \models \varphi$ whenever $\mathsf{trace}(\pi) \models \varphi$. To capture all signals satisfying MITL specification φ, we introduce a variant of IA, called timed transducer [8], [12].

Definition 1 (Timed Transducer): A timed transducer is an 8-tuple $\mathcal{B} = (X, x_0^B, \Sigma_B, \mathcal{X}_B, \mathsf{inv}_B, \xi, \eta, X_m)$, where X is a finite set of states, x_0^B is the initial state and $x_0^B \notin X$, Σ_B is an alphabet, $\mathcal{X}_B$ is a finite set of clocks, $\mathsf{inv}_B : X \to C(\mathcal{X}_B)$ is an invariant function, $\xi \subseteq X \cup \{x_0^B\} \times C(\mathcal{X}_B) \times 2^{\mathcal{X}_B} \times X$ is the transition relation, $\eta : X \cup \xi \to$

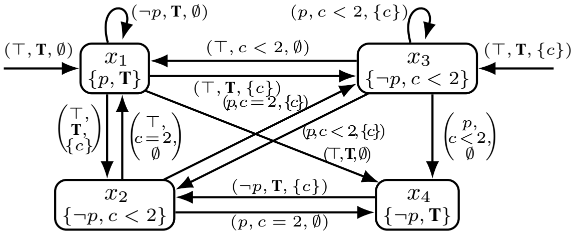

x_1 $\{p, \mathbf{T}\}$
x_2 $\{\neg p, c < 2\}$
x_3 $\{\neg p, c < 2\}$
x_4 $\{\neg p, \mathbf{T}\}$
$(\neg p, \mathbf{T}, \emptyset)$
$(p, c < 2, \{c\})$
$(\top, \mathbf{T}, \emptyset)$
$(\top, c < 2, \emptyset)$
$(\top, \mathbf{T}, \{c\})$
$(\top, \mathbf{T}, \{c\})$
$(p, c = 2, \{c\})$
$(\top, \mathbf{T}, \{c\})$
$(\top, c = 2, \emptyset)$
$(p, c < 2, \{c\})$
$(\top, \mathbf{T}, \emptyset)$
$(p, c < 2, \emptyset)$
$(\neg p, \mathbf{T}, \{c\})$
$(p, c = 2, \emptyset)$

Fig. 2: Timed transducer $\mathcal{B}_\varphi$ of specification $\varphi = \Diamond_{(0,2)} p$ with $\mathcal{AP} = \{p\}$. All states and transitions are accepting.

$BC(\Sigma_B)$ is the labeling function, where $BC(\Sigma_B)$ denotes the Boolean combinations over Σ_B, and $X_m \subseteq X \cup \xi$ is a set of accepting states and transitions.

Similar to classical IA in Equation (1), there are two types of transitions in timed transducer: time transition and discrete transition, while the discrete transition consumes a time instant in timed transducer. Specifically, a run of timed transducer $\mathcal{B}$ over a signal $\mathsf{s} \in \mathsf{Sig}(\Sigma_B)$ is an alternation of discrete and time transitions stating from the initial state (x_0^B, v_0), i.e., $\pi = (x_0^B, v_0) \xrightarrow{e_0} (x_1, v_1) \xrightarrow{t_1} (x_1, v_1 + t_1) \xrightarrow{e_1} (x_2, v_2) \cdots$, where $v_0 = \mathbf{0}_{\mathcal{X}}, e_0 = (x_0^B, g_0, \mathcal{Y}_0, x_1) \in \xi$ such that $v_0 \models g_0, v_1 = v_{0[\mathcal{Y}_0 \leftarrow 0]}$ and $\mathsf{s}(0) \models \eta(e_1)$, and for any $i = 1, 2, \cdots$, we have $t_i \in \mathbb{R}_+ \setminus \{0\}$ and (i) $e_i = (x_i, g, \mathcal{Y}, x_{i+1}) \in \xi, v_i + t_i \models g, v_{i+1} = v_{i[\mathcal{Y} \leftarrow 0]}$ and $\mathsf{s}(\sum_{k=1}^{i} t_k) \models \eta(e_i)$; (ii) for any $t' \in (0, t_i)$, $v_i + t' \models \mathsf{inv}_B(x_i)$ and $\mathsf{s}(t' + \sum_{k=1}^{i-1} t_k) \models \eta(x_i)$. An infinite signal $\mathsf{s} \in \mathsf{Sig}(\Sigma_B)$ is accepted by timed transducer $\mathcal{B}$ if the run π over s visits the state in X_m for unbounded time duration or the transition in X_m for infinite times, i.e., $\mathsf{inf}(\pi) \cap X_m \neq \emptyset$, where $\mathsf{inf}(\pi)$ contains all states in X visited by π for unbounded time duration and all transitions enabled in π for infinite number of times. We denote by $\mathcal{L}_m^\omega(\mathcal{B})$ the set of all signals accepted by $\mathcal{B}$.

Given an MITL formula φ, we can translate φ into a timed transducer $\mathcal{B}_\varphi$ with alphabet $\Sigma_B = \mathcal{AP}$ such that $\mathcal{L}_m^\omega(\mathcal{B}_\varphi) = \mathsf{signal}(\varphi)$ [8], [12]; we say such $\mathcal{B}_\varphi$ is associated with φ.

III. Time Sensitive Diagnosability

In the failure diagnosis problem, system is assumed to be subject to faults. In this paper, we use an MITL formula φ to model the time-sensitive specifications for a timed system and define faults by runs violating φ. Specifically, we denote by $\mathsf{Run}_F^\omega(G) = \{\pi \in \mathsf{Run}^\omega(G) : \pi \not\models \varphi\}$ the set of infinite faulty runs in an IA G. Then we say a finite run is faulty if any of its infinite extension is faulty and the set of finite faulty runs is denoted by $\mathsf{Run}_F^*(G) = \{\pi \in \mathsf{Run}^*(G) : \forall \pi\pi' \in \mathsf{Run}^\omega(G) \text{ s.t. } \pi\pi' \not\models \varphi\}$. Similarly, an infinite run π is normal if it satisfies φ, and the set of infinite normal runs is $\mathsf{Run}_N^\omega(G) = \{\pi \in \mathsf{Run}^\omega(G) : \pi \models \varphi\}$. A finite run is normal if it is a prefix of some infinite normal run and the set of finite normal runs is $\mathsf{Run}_N^*(G) = \mathsf{Pre}(\mathsf{Run}_N^\omega(G)) = \{\pi \in \mathsf{Run}^*(G) : \exists \pi\pi' \in \mathsf{Run}^\omega(G) \text{ s.t. } \pi\pi' \models \varphi\}$. It is obvious that the set of finite normal runs is the complement of the set of finite faulty runs, i.e., $\mathsf{Run}_N^*(G) = \mathsf{Run}^*(G) \setminus \mathsf{Run}_F^*(G)$. Then we define *Time-Sensitive Diagnosability* (TS-diagnosability) as follows:

Definition 2 (Time Sensitive Diagnosability): Let $G = (Q, Q_0, \mathcal{X}, \mathsf{inv}, E, \mathcal{AP}, L)$ be an IA with output labels $\mathcal{O}$, observation function H, and φ be an MITL formula describing the normal behaviors. We say system G is time-sensitively diagnosable (TS-diagnosable) if

$$(\forall \pi_1 \in \mathsf{Run}_F^\omega(G))(\exists \pi_1' \in \mathsf{Pre}(\pi_1))[\textsf{TS-Diag}],$$

where the diagnosis condition $\textsf{TS-Diag}$ is

$$(\forall \pi_2 \in \mathsf{Run}^*(G))[\rho_{\pi_2} = \rho_{\pi_1'} \Rightarrow \pi_2 \in \mathsf{Run}_F^*(G)].$$

TS-diagnosability says that for any infinite faulty run π_1, it has a finite prefix π_1' such that for any finite run π_2, if it has the same observation with π_1', then π_2 is a finite faulty run.

Example 2: Again, we consider IA G in Figure 1. We assume that the specification of G is given by MITL formula $\varphi = \Diamond_{(0,2)} p$. Note that there exists an infinite faulty run

$$\pi_f = \xrightarrow[\mathbf{0}_{\mathcal{X}}]{\varepsilon} (1, [0, 1]) \xrightarrow[v_1]{\delta_1} (2, (0, 1.5]) \xrightarrow[v_2]{\delta_2} (3, (0, 0.5]) \xrightarrow[v_3]{\delta_3} (5, (0, \infty)) \tag{2}$$

such that $\mathsf{trace}(\pi_f) = (\emptyset, [0, 2.5])(p, (0, 0.5])(\emptyset, (0, \infty)) \not\models \varphi$, i.e., $\pi_f \in \mathsf{Run}_F^\omega(G)$. However, there exists another run

$$\pi_n = \xrightarrow[\mathbf{0}_{\mathcal{X}}]{\varepsilon} (1, [0, 1]) \xrightarrow[v_1]{\delta_1} (2, (0, 0.5]) \xrightarrow[v_2]{\delta_2} (4, (0, 1]) \xrightarrow[v_3]{\delta_3} (6, (0, 0.5]) \xrightarrow[v_4]{\delta_4} (5, (0, \infty)), \tag{3}$$

whose timed trace is $\mathsf{trace}(\pi_n) = (\emptyset, [0, 1.5])(p, (0, 1]) (\emptyset, (0, \infty)) \models \varphi$, i.e., $\pi_n \in \mathsf{Run}_N^\omega(G)$, such that the observations of π_f and π_n are the same, i.e, $\rho_{\pi_f} = \rho_{\pi_n} = (o_1, [0, 1])(o_2, (0, 1.5])(o_3, (0, 0.5])(o_2, (0, \infty))$. Therefore, for any prefix $\pi_f' \in \mathsf{Pre}(\pi_f)$, there exists a finite normal run $\pi_n' \in \mathsf{Pre}(\pi_n)$ such that $\rho_{\pi_f'} = \rho_{\pi_n'}$. By Definition 2, system G is not TS-diagnosable.

IV. Verification of Time Sensitive Diagnosability

A. Constrained System

Given an IA G and a specification φ, to verify TS-diagnosability, we first recognize all normal runs and faulty runs, respectively. Recall that for any MITL formula φ, we can obtain a timed transducer $\mathcal{B}_\varphi$ such that $\mathcal{L}_m^\omega(\mathcal{B}_\varphi) = \mathsf{signal}(\varphi)$. To capture all normal runs, we construct the *positive constrained system.*

Definition 3 (Positive Constrained System): Given an IA $G = (Q, Q_0, \mathcal{X}, \mathsf{inv}, E, \mathcal{AP}, L)$ and a timed transducer $\mathcal{B}_\varphi = (X^p, x_0^p, \mathcal{AP}, \mathcal{X}_B^p, \mathsf{inv}_B^p, \xi^p, \eta^p, X_m^p)$, the positive constrained system is defined as a new-tuple

$$T_p = (Q_p, Q_{p,0}, \mathcal{X}_p, \mathsf{inv}_p, E_p, Q_{p,m}),$$

where

- $Q_p = Q_Y \cup Q_Z$ is the set of discrete states, where $Q_Y = Q \times X^p$ and $Q_Z = Q \times \xi^p$;
- $Q_{p,0} = \{(q_0, (x_0^p, g_B, \mathcal{Y}_B, x_1)) \in Q_0 \times \xi^p : L(q_0) \models \eta^p((x_0^p, g_B, \mathcal{Y}_B, x_1))\}$ is the set of initial discrete states;
- $\mathcal{X}_p = \mathcal{X} \cup \mathcal{X}_B^p \cup \{c_z\}$ is a finite set of clocks, where $c_z \notin \mathcal{X} \cup \mathcal{X}_B^p$ is a new clock;

- $\mathsf{inv}_p : Q_p \to C(\mathcal{X}_p)$ is the invariant function defined by: for any state $q_p = (q, x) \in Q_p$, we have

$$\mathsf{inv}_p(q_p) = \begin{cases} \mathsf{inv}(q) \wedge \mathsf{inv}^p_B(x), & \text{if } q_p \in Q_Y \\ \mathsf{inv}(q) \wedge c_z = 0, & \text{if } q_p \in Q_Z \end{cases};$$

- $E_p \subseteq Q_p \times C(\mathcal{X}_p) \times 2^{\mathcal{X}_p} \times Q_p$ is the transition relation defined by:
  - for any $(q, x'), (q', x') \in Q_Y$ and $(q, (x, g_B, \mathcal{Y}_B, x')) \in Q_Z$, we have

$$L(q) \models \eta^p(x') \Rightarrow$$
$$((q, (x, g_B, \mathcal{Y}_B, x')), \mathsf{inv}(q), \mathcal{Y}_B, (q, x')) \in E_p,$$
$$L(q') \models \eta^p(x') \wedge (q, g, \mathcal{Y}, q') \in E \Rightarrow$$
$$((q, (x, g_B, \mathcal{Y}_B, x')), g, \mathcal{Y} \cup \mathcal{Y}_B, (q', x')) \in E_p;$$

  - for any $(q, x), (q', x) \in Q_Y$, we have

$$L(q') \models \eta^p(x) \wedge (q, g, \mathcal{Y}, q') \in E \Rightarrow$$
$$((q, x), g \wedge \mathsf{inv}^p_B(x), \mathcal{Y}, (q', x)) \in E_p;$$

  - for any $(q, x) \in Q_Y$ and $(q, (x, g_B, \mathcal{Y}_B, x'))$, $(q', (x, g_B, \mathcal{Y}_B, x')) \in Q_Z$, we have

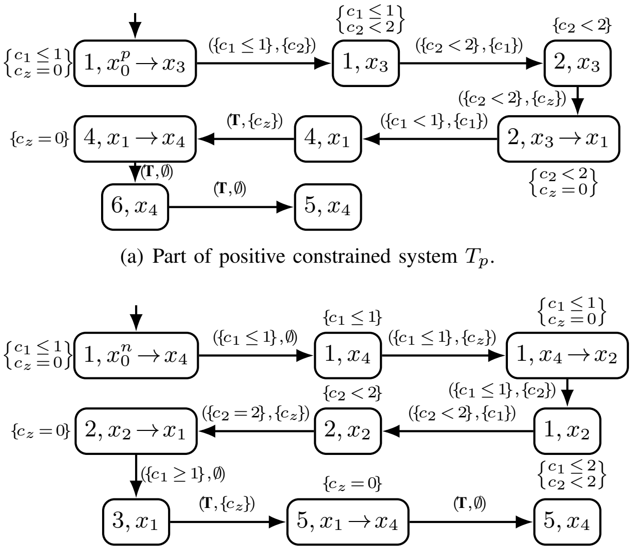
$$L(q) \models \eta^p((x, g_B, \mathcal{Y}_B, x')) \Rightarrow ((q, x), \mathsf{inv}(q) \wedge$$
$$g_B, \{c_z\}, (q, (x, g_B, \mathcal{Y}_B, x'))) \in E_p,$$
$$L(q') \models \eta^p((x, g_B, \mathcal{Y}_B, x')) \wedge (q, g, \mathcal{Y}, q') \in E \Rightarrow$$
$$((q, x), g \wedge g_B, \mathcal{Y} \cup \{c_z\}, (q', (x, g_B, \mathcal{Y}_B, x'))) \in E_p;$$

- $Q_{p,m} = \{(q, x) \in Q_p : x \in X^p_m\}$ is accepting states set.

Intuitively, the positive constrained system is constructed by synchronizing the original system G with the timed transducer $\mathcal{B}_\varphi$ according to internal behaviors. Specifically, each state $(q, x) \in Q_Y$ is a pair of state in system G and state in $\mathcal{B}_\varphi$ matching the time interval when the label of q satisfies the label of x, while state $(q, e) \in Q_Z$ is a pair of state in system G and transition in $\mathcal{B}_\varphi$ representing the time instant such that the label of q satisfies the label of e. Each transition in $\mathcal{B}_\varphi$ consumes one time instant. To match the time elapsing, we actually regard each transition in $\mathcal{B}_\varphi$ as a new discrete state with invariant $c_z = 0$.

A discrete state $(q, x) \in Q_p$ is accepting if its second component is accepting in $\mathcal{B}_\varphi$. Let $\pi_p = \xrightarrow[v_0]{\delta_0} ((q_0, x_0), \Delta_0) \xrightarrow[v_1]{\delta_1} ((q_1, x_1), \Delta_1) \xrightarrow[v_2]{\delta_2} \cdots$ be an infinite run of T_p. We say that π is accepted by T_p if it visits discrete state $(q, x) \in Q_{p,m}$ for infinite number of times. We denote by $\mathrm{Run}_m(T_p)$ all infinite runs accepted by T_p. Given a timed sequence $s = ((q^l_0, q^r_0), \Delta_0)((q^l_1, q^r_1), \Delta_1), \cdots$, we denote the left and right timed state sequence of it by $s^l = (q^l_0, \Delta_0)(q^l_1, \Delta_1) \cdots$ and $s^r = (q^r_0, \Delta_0)(q^r_1, \Delta_1) \cdots$, respectively. By construction of T_p, an infinite run $\pi_p \in \mathrm{Run}^\omega(T_p)$ is accepted by T_p if there exists an infinite normal run $\pi \in \mathsf{Run}^\omega(G)$ such that $s^l_{\pi_p} = s_\pi$ and $\mathsf{trace}(\pi) \models \varphi$. For any discrete state $(q, x) \in Q_p$, we define the observation of it as $H_p((q, x)) = H(q)$. Then, we have the following equivalence

$$\exists \pi \in \mathsf{Run}^\omega_N(G) \Leftrightarrow$$
$$\exists \pi_p \in \mathsf{Run}_m(T_p), s^l_{\pi_p} = s_\pi \wedge \rho_{\pi_p} = \rho_\pi. \quad (4)$$



(a) Part of positive constrained system T_p.

(b) Part of negative constrained system T_n.

Fig. 3: Constrained system T_p and T_n in (a) and (b), respectively. We represent each state $(q, (x, g_B, \mathcal{Y}_B, x')) \in Q_Z$ by $(q, x \to x')$.

On the other hand, to recognize all faulty runs, we can obtain the *negative constrained system* $T_n = (Q_n, Q_{n,0}, \mathcal{X}_n, \mathsf{inv}_n, E_n, Q_{n,m})$ based on the time transducer $\mathcal{B}_{\neg\varphi} = (X^n, x^n_0, \mathcal{AP}, \mathcal{X}^n_B, \mathsf{inv}^n_B, \xi^n, \eta^n, X^n_m)$ in the similar way.

To recognize finite normal runs, we first translate T_p into its region automaton $T^R_p = (Q^R_p, Q^R_{p,0}, \Sigma^R_p, E^R_p, Q^R_{p,m})$, where $Q^R_{p,m} \subseteq Q^R_p$ is the set of accepting states defined by $Q^R_{p,m} = \{(q, r) \in Q^R_p : q \in Q_{p,m}\}$. An infinite run in T^R_p is accepted if it visits states in $Q^R_{p,m}$ infinitely and we denote by $\mathsf{Run}^R_m(T_p)$ the set of all infinite runs accepted by T^R_p. By Equation (4), we know that finite run $\pi \in \mathsf{Run}^*(G)$ is normal iff there exists a finite run $\pi_p \in \mathsf{Run}^*(T_p)$ such that $s^l_{\pi_p} = s_\pi$ and π_p can be extended to an infinite run visiting states in $Q_{p,m}$ infinitely, i.e., $\pi_p \in \mathsf{Pre}(\mathsf{Run}_m(T_p))$. To recognize all runs in $\mathsf{Pre}(\mathsf{Run}_m(T_p))$, we say a state $q_p = (q, r) \in Q^R_p$ is *normal feasible* if $\mathsf{Run}^R_m(T^R_p(q_p)) \neq \emptyset$, where $T^R_p(q_p) = (Q^R_p, \{q_p\}, \Sigma^R_p, E^R_p, Q^R_{p,m})$ is equivalent to T^R_p except that the initial state is set to q_p. Then, we denote by $Q^R_{feas} \subseteq Q^R_p$ the set of all normal feasible states in T^R_p. We have the following equivalence

$$(\pi \in \mathsf{Pre}(\mathsf{Run}_m(T_p))) \Leftrightarrow (\exists \pi^R \in \mathsf{Run}^R(T^R_p))$$
$$(\mathsf{unt}(\pi) = \mathsf{unt}^R(\pi^R) \wedge \mathsf{last}_R(\pi^R) \in Q^R_{feas}). \quad (5)$$

Based on the normal feasible state set Q^R_{feas}, we say a state $(q, v) \in Q_p \times \mathcal{V}_{\mathcal{X}_p}$ is normal feasible if there exists a state (q, r) in Q^R_{feas} such that $v \in r$. We denote by $Q_{feas} = \{(q, v) \in Q_p \times \mathcal{V}_{\mathcal{X}_p} : \exists (q, r) \in Q^R_{feas}, v \in r\}$ the set of all feasible states in T_p. Then, for any run $\pi \in \mathsf{Run}^*(T_p)$, we have the following equivalence

$$\pi \in \mathsf{Pre}(\mathsf{Run}_m(T_p)) \Leftrightarrow \mathsf{last}(\pi) \in Q_{feas}. \quad (6)$$

Example 3: Still, we consider IA G in Figure 1 with specification $\varphi = \Diamond_{(0,2)} p$. We translate φ to timed transducer $\mathcal{B}_\varphi$ as shown in Figure 2, where we omit the initial state x^p_0 and all states and transitions are accepting. The timed

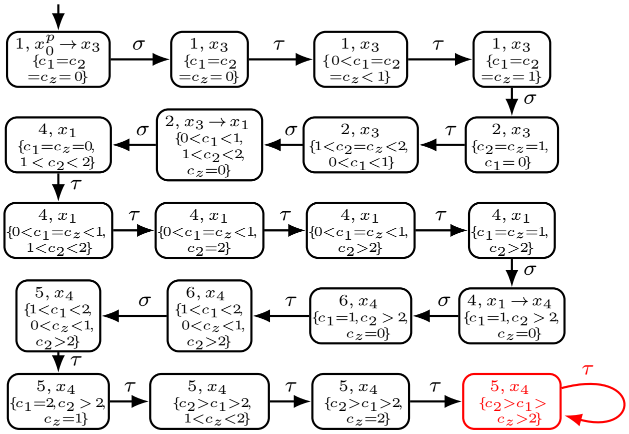

$1, x_0^p \to x_3$ $\{c_1{=}c_2{=}c_z{=}0\}$
σ
$1, x_3$ $\{c_1{=}c_2{=}c_z{=}0\}$
τ
$1, x_3$ $\{0{<}c_1{=}c_2{=}c_z{<}1\}$
τ
$1, x_3$ $\{c_1{=}c_2{=}c_z{=}1\}$
σ
$2, x_3$ $\{c_2{=}c_z{=}1, c_1{=}0\}$
τ
$2, x_3$ $\{1{<}c_2{=}c_z{<}2, 0{<}c_1{<}1\}$
σ
$2, x_3 \to x_1$ $\{0{<}c_1{<}1, 1{<}c_2{<}2, c_z{=}0\}$
σ
$4, x_1$ $\{c_1{=}c_z{=}0, 1{<}c_2{<}2\}$
τ
$4, x_1$ $\{0{<}c_1{=}c_z{<}1, 1{<}c_2{<}2\}$
τ
$4, x_1$ $\{0{<}c_1{=}c_z{<}1, c_2{=}2\}$
τ
$4, x_1$ $\{0{<}c_1{=}c_z{<}1, c_2{>}2\}$
τ
$4, x_1$ $\{c_1{=}c_z{=}1, c_2{>}2\}$
σ
$4, x_1 \to x_4$ $\{c_1{=}1, c_2>2, c_z{=}0\}$
σ
$6, x_4$ $\{c_1{=}1, c_2>2, c_z{=}0\}$
τ
$6, x_4$ $\{1{<}c_1{<}2, 0{<}c_z{<}1, c_2{>}2\}$
σ
$5, x_4$ $\{1{<}c_1{<}2, 0{<}c_z{<}1, c_2{>}2\}$
τ
$5, x_4$ $\{c_1{=}2, c_2>2, c_z{=}1\}$
τ
$5, x_4$ $\{c_2{>}c_1{>}2, 1{<}c_z{<}2\}$
τ
$5, x_4$ $\{c_2{>}c_1{>}2, c_z{=}2\}$
τ
$5, x_4$ $\{c_2{>}c_1{>}c_z{>}2\}$
τ

Fig. 4: Part of region automaton of positive constrained system T_p^R, where all states are normal feasible.

transducer $\mathcal{B}_{\neg\varphi}$ is exactly same as $\mathcal{B}_\varphi$ except that we replace edge $(\top, \mathbf{T}, \emptyset)$ from x_0^p to x_1 and edge $(\top, \mathbf{T}, \{c\})$ from x_0^p to x_3 in $\mathcal{B}_\varphi$ with edge $(\top, \mathbf{T}, \{c\})$ from x_0^p to x_2 and edge $(\top, \mathbf{T}, \emptyset)$ from x_0^p to x_4, respectively, which are omitted. Based on $\mathcal{B}_\varphi$ and $\mathcal{B}_{\neg\varphi}$, we construct positive constrained system T_p and negative positive constrained system T_n, which are partially shown in Figure 3(a) and Figure 3(b), respectively, with all states accepting. The part of T_p depicted in Figure 3(a) contains the normal run π_n in Equation (3) and the faulty run π_f in Equation (2) is embedded in the part of T_n shown in Figure 3(b). Then, we construct the region automaton T_p^R and the part that abstracts the part T_p in Figure 3(a) is depicted in Figure 4, where all states are accepting. Since, starting from each state in Figure 4, there is an infinite run that can reach the final state $((5, x_4), \{c_1 > 2, c_2 > 2, c_z > 2\})$ as highlighted by red color and can repeat the self loop infinitely, all states are normal feasible.

B. Verification System

Given IA G and MITL specification φ, we can obtain $T_p = (Q_p, Q_{p,0}, \mathcal{X}_p, \mathsf{inv}_p, E_p,\ Q_{p,m})$ and $T_n = (Q_n, Q_{n,0}, \mathcal{X}_n, \mathsf{inv}_n, E_n, Q_{n,m})$. Then we construct the verification system

$$V = (Q_V, Q_{V,0}, \mathcal{X}_V, \mathsf{inv}_V, E_V)$$

where

- $Q_V = \{(q_n, q_p) \in Q_n \times Q_p : H_n(q_n) = H_p(q_p)\}$ is the set of discrete states;
- $Q_{V,0} = \{(q_n, q_p) \in Q_{n,0} \times Q_{p,0} : H_n(q_n) = H_p(q_p)\}$ is the set of initial discrete states;
- $\mathcal{X}_V = \mathcal{X}_n \cup \mathcal{X}_p$ is a finite set of clocks;
- $\mathsf{inv}_V : Q_V \to C(\mathcal{X}_V)$ is the invariant function defined by: for any state $q_V = (q_n, q_p) \in Q_V$, $\mathsf{inv}_V(q_V) = \mathsf{inv}_n(q_n) \wedge \mathsf{inv}_p(q_p)$;
- $E_V \subseteq Q_V \times C(\mathcal{X}_V) \times 2^{\mathcal{X}_V} \times Q_V$ is the transition relation, defined by:
  - $(q_n, g_n, \mathcal{Y}_n, q_n') \in E_n \Rightarrow ((q_n, q_p), g_n, \mathcal{Y}_n, (q_n', q_p)) \in E_V$
  - $(q_p, g_p, \mathcal{Y}_p, q_p') \in E_p \Rightarrow ((q_n, q_p), g_p, \mathcal{Y}_p, (q_n, q_p')) \in E_V$
  - $(q_n, g_n, \mathcal{Y}_n, q_n') \in E_n \wedge (q_p, g_p, \mathcal{Y}_p, q_p') \in E_p \Rightarrow ((q_n, q_p), g_n \wedge g_p, \mathcal{Y}_n \cup \mathcal{Y}_p, (q_n', q_p')) \in E_V$.

Essentially, verification system V tracks every pair of runs in T_n and T_p that have the same observation. For any state

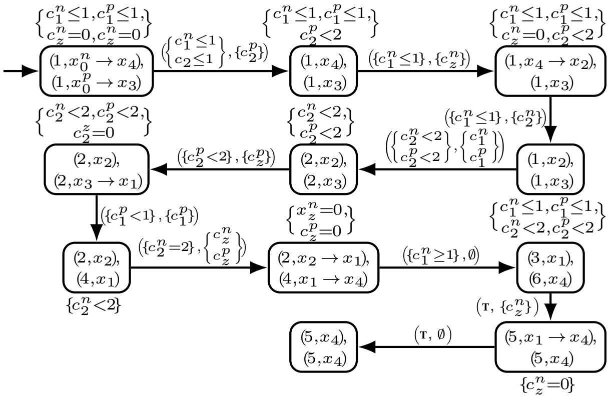

$\left\{\begin{matrix} c_1^n{\le}1, c_1^p{\le}1, \\ c_z^n{=}0, c_z^n{=}0 \end{matrix}\right\}$
$(1, x_0^n \to x_4),$ $(1, x_0^p \to x_3)$
$\left(\left\{\begin{matrix} c_1^n{\le}1 \\ c_2{\le}1 \end{matrix}\right\}, \{c_2^p\}\right)$
$\left\{\begin{matrix} c_1^n{\le}1, c_1^p{\le}1, \\ c_2^p{<}2 \end{matrix}\right\}$
$(1, x_4),$ $(1, x_3)$
$(\{c_1^n{\le}1\}, \{c_z^n\})$
$\left\{\begin{matrix} c_1^n{\le}1, c_1^p{\le}1, \\ c_z^n{=}0, c_2^p{<}2 \end{matrix}\right\}$
$(1, x_4 \to x_2),$ $(1, x_3)$
$(\{c_1^n{\le}1\}, \{c_2^n\})$
$(1, x_2),$ $(1, x_3)$
$\left(\left\{\begin{matrix} c_2^n{<}2 \\ c_2^p{<}2 \end{matrix}\right\}, \left\{\begin{matrix} c_1^n \\ c_1^p \end{matrix}\right\}\right)$
$\left\{\begin{matrix} c_2^n{<}2, \\ c_2^p{<}2 \end{matrix}\right\}$
$(2, x_2),$ $(2, x_3)$
$(\{c_2^p{<}2\}, \{c_z^p\})$
$\left\{\begin{matrix} c_2^n{<}2, c_2^p{<}2, \\ c_2^z{=}0 \end{matrix}\right\}$
$(2, x_2),$ $(2, x_3 \to x_1)$
$(\{c_1^p{<}1\}, \{c_1^p\})$
$(2, x_2),$ $(4, x_1)$
$\{c_2^n{<}2\}$
$\left(\{c_2^n{=}2\}, \left\{\begin{matrix} c_z^n \\ c_z^p \end{matrix}\right\}\right)$
$\left\{\begin{matrix} x_z^n{=}0, \\ c_z^p{=}0 \end{matrix}\right\}$
$(2, x_2 \to x_1),$ $(4, x_1 \to x_4)$
$(\{c_1^n{\ge}1\}, \emptyset)$
$\left\{\begin{matrix} c_1^n{\le}1, c_1^p{\le}1, \\ c_2^n{<}2, c_2^p{<}2 \end{matrix}\right\}$
$(3, x_1),$ $(6, x_4)$
$(\tau, \{c_z^n\})$
$(5, x_1 \to x_4),$ $(5, x_4)$
$\{c_z^n{=}0\}$
$(\tau, \emptyset)$
$(5, x_4),$ $(5, x_4)$

Fig. 5: Part of verification system V.

$(q_n, q_p) \in Q_V$, we denote by $H_V((q_n, q_p)) = H_n(q_n) = H_p(q_p)$ the output of this state. Then the verification system has the following properties: (i) For any π in V with last state $\mathsf{last}(\pi) = ((q_n, q_p), v)$, there exist two runs π_n in T_n and π_p in T_p such that $s_{\pi_n} = s_\pi^l, s_{\pi_p} = s_\pi^r, \rho_{\pi_n} = \rho_{\pi_p} = \rho_\pi$, and the last states of π_n and π_p satisfy $\mathsf{last}(\pi_n) = (q_n, v_n), \mathsf{last}(\pi_p) = (q_p, v_p), v_n = v_{\mathcal{X}_n}$ and $v_p = v_{\mathcal{X}_p}$; (ii) For any f $\pi_n \in \mathsf{Run}(T_n)$ and $\pi_p \in \mathsf{Run}(T_p)$ with last states $\mathsf{last}(\pi_n) = (q_n, v_n)$ and $\mathsf{last}(\pi_p) = (q_p, v_p)$, if they have the same observation, there exists a run $\pi \in \mathsf{Run}(V)$ such that $s_{\pi_n} = s_\pi^l, s_{\pi_p} = s_\pi^r, \rho_{\pi_n} = \rho_{\pi_p} = \rho_\pi$ and the last state of π is $\mathsf{last}(\pi) = ((q_n, q_p), v)$ satisfying $v_{\mathcal{X}_n} = v_n$ and $v_{\mathcal{X}_p} = v_p$.

C. Checking TS-Diagnosability

Now, we present how to verify TS-diagnosability using the verification system. For a verification system $V = (Q_V, Q_{V,0}, \mathcal{X}_V, \mathsf{inv}_V, E_V)$, we first translate it into region automaton $V^R = (Q_V^R, Q_{0,V}^R, \Sigma_V^R, E_V^R, Q_m^R)$, where $Q_m^R \subseteq Q_V^R$ is the set of accepting states defined by:

$$\begin{aligned} Q_m^R = \{&((q_n, q_p), r) \in Q_V^R : q_n \in Q_{n,m} \wedge \\ &(\exists (q_p, r') \in Q_{feas}^R)(r(\mathcal{X}_p) = r')\}. \end{aligned} \tag{7}$$

Intuitively, a state $q_V^R = ((q_n, q_p), r) \in Q_m^R$ is accepting if (i) q_n is an accepting state in negative constrained system; and (ii) the second component of discrete state q_p and the clock region r can be reduced to a normal feasible state $(q_p, r(\mathcal{X}_p)) \in Q_{feas}^R$. We are now ready to present the the necessary and sufficient conditions for TS-diagnosability.

Theorem 1: System G is not TS-diagnosable w.r.t. observation function H and MITL specification φ, if and only if, in the region automaton V^R of verification system V, there exists a reachable cycle

$$\pi = q_1^R \xrightarrow{\lambda_1} q_2^R \xrightarrow{\lambda_2} \cdots \xrightarrow{\lambda_{n-1}} q_n^R$$

such that $q_i^R \in Q_m^R$ and $\lambda_j = \tau$, for some $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, n-1\}$.

Intuitively, since π is a reachable cycle, we can find an infinite run $\pi' \in \mathsf{Run}^R(V^R)$ starting from some initial state in $Q_{0,V}^R$ and going through cycle π infinitely. Then, the condition $\lambda_j = \tau$ ensures that there exists an infinite run $\pi_V \in \mathsf{Run}(V)$ embedded in π'; and the condition $q_i^R \in Q_m^R$ ensures that (i) there is an infinite run $\pi_n \in \mathsf{Run}_m(T_n)$

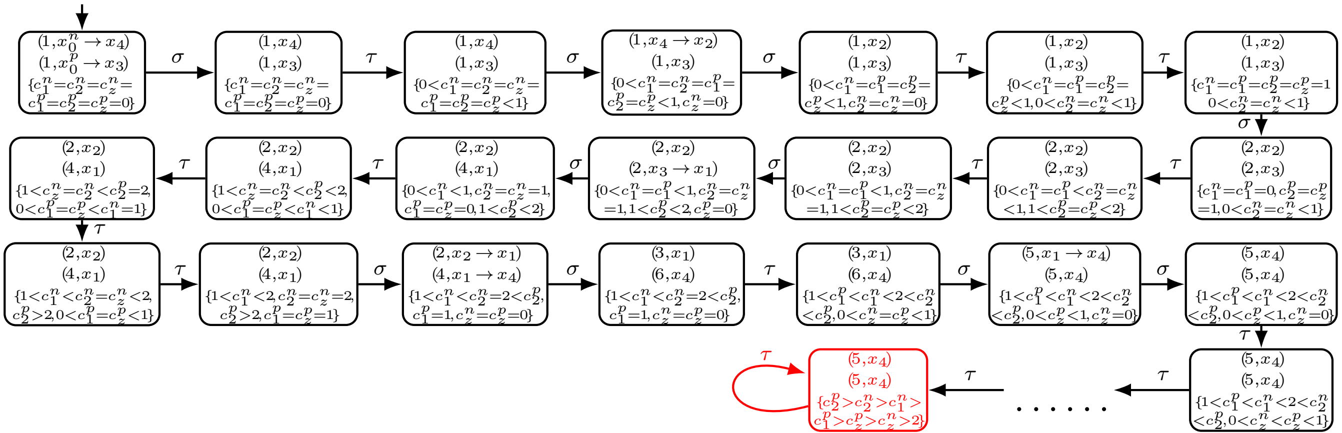

$(1,x_0^n \to x_4)$ $(1,x_0^p \to x_3)$ $\{c_1^n=c_2^n=c_z^n=c_1^p=c_2^p=c_z^p=0\}$
σ
$(1,x_4)$ $(1,x_3)$ $\{c_1^n=c_2^n=c_z^n=c_1^p=c_2^p=c_z^p=0\}$
τ
$(1,x_4)$ $(1,x_3)$ $\{0<c_1^n=c_2^n=c_z^n=c_1^p=c_2^p=c_z^p<1\}$
σ
$(1,x_4 \to x_2)$ $(1,x_3)$ $\{0<c_1^n=c_2^n=c_1^p=c_2^p=c_z^p<1,c_z^n=0\}$
σ
$(1,x_2)$ $(1,x_3)$ $\{0<c_1^n=c_1^p=c_2^p=c_z^p<1,c_2^n=c_z^n=0\}$
τ
$(1,x_2)$ $(1,x_3)$ $\{0<c_1^n=c_1^p=c_2^p=c_z^p<1,0<c_2^n=c_z^n<1\}$
τ
$(1,x_2)$ $(1,x_3)$ $\{c_1^n=c_1^p=c_2^p=c_z^p=1$ $0<c_2^n=c_z^n<1\}$
σ
$(2,x_2)$ $(2,x_3)$ $\{c_1^n=c_1^p=0,c_2^p=c_z^p=1,0<c_2^n=c_z^n<1\}$
τ
$(2,x_2)$ $(2,x_3)$ $\{0<c_1^n=c_1^p<c_2^n=c_z^n<1,1<c_2^p=c_z^p<2\}$
τ
$(2,x_2)$ $(2,x_3)$ $\{0<c_1^n=c_1^p<1,c_2^n=c_z^n=1,1<c_2^p=c_z^p<2\}$
σ
$(2,x_2)$ $(2,x_3 \to x_1)$ $\{0<c_1^n=c_1^p<1,c_2^n=c_z^n=1,1<c_2^p<2,c_z^p=0\}$
σ
$(2,x_2)$ $(4,x_1)$ $\{0<c_1^n<1,c_2^n=c_z^n=1,c_1^p=c_z^p=0,1<c_2^p<2\}$
τ
$(2,x_2)$ $(4,x_1)$ $\{1<c_z^n=c_2^n<c_2^p<2,0<c_1^p=c_z^p<c_1^n<1\}$
τ
$(2,x_2)$ $(4,x_1)$ $\{1<c_z^n=c_2^n<c_2^p=2,0<c_1^p=c_z^p<c_1^n=1\}$
τ
$(2,x_2)$ $(4,x_1)$ $\{1<c_1^n<c_2^n=c_z^n<2,c_2^p>2,0<c_1^p=c_z^p<1\}$
τ
$(2,x_2)$ $(4,x_1)$ $\{1<c_1^n<2,c_2^n=c_z^n=2,c_2^p>2,c_1^p=c_z^p=1\}$
σ
$(2,x_2 \to x_1)$ $(4,x_1 \to x_4)$ $\{1<c_1^n<c_2^n=2<c_2^p,c_1^p=1,c_z^n=c_z^p=0\}$
σ
$(3,x_1)$ $(6,x_4)$ $\{1<c_1^n<c_2^n=2<c_2^p,c_1^p=1,c_z^n=c_z^p=0\}$
τ
$(3,x_1)$ $(6,x_4)$ $\{1<c_1^p<c_1^n<2<c_2^n<c_2^p,0<c_z^n=c_z^p<1\}$
σ
$(5,x_1 \to x_4)$ $(5,x_4)$ $\{1<c_1^p<c_1^n<2<c_2^n<c_2^p,0<c_z^p<1,c_z^n=0\}$
σ
$(5,x_4)$ $(5,x_4)$ $\{1<c_1^p<c_1^n<2<c_2^n<c_2^p,0<c_z^p<1,c_z^n=0\}$
τ
$(5,x_4)$ $(5,x_4)$ $\{1<c_1^p<c_1^n<2<c_2^n<c_2^p,0<c_z^n<c_z^p<1\}$
τ
. . . . . .
τ
$(5,x_4)$ $(5,x_4)$ $\{c_2^p>c_2^n>c_1^n>c_1^p>c_z^p>c_z^n>2\}$
τ

Fig. 6: Part of region automaton of verification system V^R. The cycle highlighted in red contains event τ and an accepting state $(((5,x_4),(5,x_4)),\{c_2^p>c_2^n>c_1^n>c_1^p>c_z^p>c_z^n>2\})$.

such that $s^l_{\pi_V} = s_{\pi_n}$; and (ii) there is an infinite run $\pi_p \in \mathsf{Run}^\omega(T_p)$ such that $s^r_{\pi_V} = s_{\pi_p}$ and any prefix $\pi'_p \in \mathsf{Pre}(\pi_p)$ can be extended to an accepting run in T_p, i.e., $\pi'_p \in \mathsf{Pre}(\mathsf{Run}_m(T_p))$. Moreover, the construction of V ensures π_n and π_p have the same output. Therefore, the existence of such a cycle falsifies TS-diagnosability.

Example 4: Still consider G in Figure 1 with the same setting in Example 2, which is not TS-diagnosable subject to specification $\varphi = \Diamond_{(0,2)}p$. Based on T_p and T_n, we construct the verification structure V. We partially depict V in Figure 5, which tracks part of T_p in Figure 3(a) and T_n Figure 3(b). Then, we obtain the region automaton V^R, partially shown in Figure 6, where all states are accepting. Here we can find a run reaching state $(((5,x_4),(5,x_4)),\{c_2^p > c_2^n > c_1^n > c_1^p > c_z^p > c_z^n > 2\})$ that has a self-loop enabled by event τ, as highlighted by red color. Note that $(5,x_4)$ is an accepting state in T_n, i.e., $(5,x_4) \in Q_{n,m}$. Let $r = \{c_2^p > c_2^n > c_1^n > c_1^p > c_z^p > c_z^n > 2\}$ and $r' = r(\mathcal{X}_p) = \{c_2^p > c_1^p > c_z^p > 2\}$. There is a state $((5,x_4),r') \in Q^R_{feas}$ as highlighted by red color in Figure 4. That is, there exists a reachable cycle satisfying all conditions in Theorem 1 and the system G is not TS-diagnosable.

V. Conclusion

In this paper, we investigated the failure diagnosis problem for timed discrete-event system with MITL specification. In contrast to existing works, we considered time-sensitive failures for timed systems formalized by MITL formulae under continuous semantics. We proposed a new notion called TS-diagnosability to capture the capability of determine failures in finite time. We also provided necessary and sufficient conditions for the verification of TS-diagnosability.

References

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

[2] R. Alur, T. Feder, and T. Henzinger. The benefits of relaxing punctuality. *Journal of the ACM*, 43(1):116–146, 1996.

[3] F. Basile, M. Cabasino, and C. Seatzu. Diagnosability analysis of labeled time Petri net systems. *IEEE Trans. Automatic Control*, 62(3):1384–1396, 2016.

[4] P. Bouyer, F. Chevalier, and D. D'Souza. Fault diagnosis using timed automata. In *8th International Conference on Foundations of Software Science and Computation Structures*, pages 219–233, 2005.

[5] L. Carvalho, M. Moreira, and J. Basilio. Comparative analysis of related notions of robust diagnosability of discrete-event systems. *Annual Reviews in Control*, 51:23–36, 2021.

[6] J. Chen and R. Kumar. Fault detection of discrete-time stochastic systems subject to temporal logic correctness requirements. *IEEE Trans. Automation Science and Engineering*, 12(4):1369–1379, 2015.

[7] W. Dong, X. Yin, and S. Li. A uniform framework for diagnosis of discrete-event systems with unreliable sensors using linear temporal logic. *IEEE Transactions on Automatic Control*, 2023.

[8] T. Ferrere, O. Maler, D. Ničković, and A. Pnueli. From real-time logic to timed automata. *Journal of the ACM*, 66(3):1–31, 2019.

[9] H. Gougam, Y. Pencolé, and A. Subias. Diagnosability analysis of patterns on bounded labeled prioritized Petri nets. *Discrete Event Dynamic Systems*, 27:143–180, 2017.

[10] T. Jéron, H. Marchand, S. Pinchinat, and M. Cordier. Supervision patterns in discrete event systems diagnosis. In *8th International Workshop on Discrete Event Systems*, pages 262–268, 2006.

[11] S. Lafortune, F. Lin, and C. Hadjicostis. On the history of diagnosability and opacity in discrete event systems. *Annual Reviews in Control*, 45:257–266, 2018.

[12] L. Lindemann, G. Pappas, and D. Dimarogonas. Reactive and risk-aware control for signal temporal logic. *IEEE Trans. Automatic Control*, 67(10):5262–5277, 2021.

[13] Z. Ma, X. Yin, and Z. Li. Marking diagnosability verification in labeled petri nets. *Automatica*, 131:109713, 2021.

[14] N. Ran, T. Li, Z. He, and C. Seatzu. Codiagnosability enforcement in labeled Petri nets. *IEEE TAC*, 68(4):2436–2443, 2023.

[15] N. Ran, H. Su, A. Giua, and C. Seatzu. Codiagnosability analysis of bounded Petri nets. *IEEE TAC*, 63(4):1192–1199, 2018.

[16] S. Takai. A general framework for diagnosis of discrete event systems subject to sensor failures. *Automatica*, 129:109669, 2021.

[17] S. Tripakis. Verifying progress in timed systems. In *ARTS*, volume 1601, pages 299–314, 1999.

[18] S. Tripakis. Fault diagnosis for timed automata. In *International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*, pages 205–221. Springer, 2002.

[19] G. Viana, M. Alves, and J. Basilio. Codiagnosability of networked discrete event systems with timing structure. *IEEE Trans. Automatic Control*, 67(8):3933–3948, 2022.

[20] X. Yin, J. Chen, Z. Li, and S. Li. Robust fault diagnosis of stochastic discrete event systems. *IEEE Trans. Automatic Control*, 64(10):4237–4244, 2019.

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

[22] X. Yin and S. Lafortune. On the decidability and complexity of diagnosability for labeled Petri nets. *IEEE Trans. Automatic Control*, 62(11):5931–5938, 2017.

[23] S. Zad, R. Kwong, and W. Wonham. Fault diagnosis in discrete-event systems: Incorporating timing information. *IEEE Trans. Automatic Control*, 50(7):1010–1015, 2005.