2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Verification of Approximate Prognosability via Barrier Certificates

Weijie Dong, Bingzhuo Zhong, Xiang Yin and Majid Zamani

2024 IEEE 63rd Conference on Decision and Control (CDC) | 979-8-3503-1633-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/CDC56724.2024.10886053

***Abstract*— In this paper, we investigate the verification of *approximate prognosability* for discrete-time control systems with continuous state set in the context of fault prognosis. Existing works on this topic rely on constructing finite abstractions, which lead to significant computation burden. To address this challenge, we propose an abstraction-free method via barrier certificates. Specifically, we consider a notion of so-called approximate (M, δ)-prognosability requiring that every fault, characterized by entering a fault region, can be predicted before its occurrence under observation precision δ and once an alarm is issued, fault will occur for sure within M time instants. Then, we propose a verification scheme based on an M-deterministic finite automaton over an augmented system of the original system. Then, we reduce the verification of (the lack of) approximate (M, δ)-prognosability to a safety verification problem, which can be checked effectively by barrier certificates. Furthermore, a counter-example guided inductive synthesis framework is proposed to compute these barrier certificates.**

I. Introduction

Cyber-physical systems (CPS), such as power systems and intelligent transportation, are critical infrastructures where faults can potentially lead to catastrophic disasters. In severe scenarios, merely diagnosing and isolating fault occurrences may not suffice to safeguard CPS, as they could occur too late, resulting in unrecoverable damages. Therefore, fault prediction or *fault prognosis* is crucial to prevent faults from causing significant damages.

In the context of model-based fault prognosis, a key challenge lies in determining whether the underlying dynamical system is *prognosable*. This notion characterizes whether the information flow generated by the system is sufficient to support the prognostic task. The notion of prognosability was first introduced in [7] within the context of discrete-event systems, which represent an important class of CPSs characterized by discrete state sets and event-triggered dynamics. Since its introduction, prognosability concept have been investigated in many literature. We refer the interested readers to some recent works [1]–[3], [10], [14], [16], [25], [26] and the survey paper [24].

This work was supported by the National Natural Science Foundation of China (61803259, 61833012), Guangzhou-HKUST(GZ) Joint Funding Program(Grant No.2023A03J0008), Education Bureau of Guangzhou Municipality, and NSF CPS Frontier project CNS-2111688.

W. Dong and X. Yin are with the Department of Automation, Shanghai Jiao Tong University, China. E-mail: {wjd_dollar,yinxiang}@sjtu.edu.cn. B. Zhong is with the Thrust of Artificial Intelligence, Information Hub, Hong Kong University of Science and Technology (Guangzhou), China. E-mail: bingzhuoz@hkust-gz.edu.cn. M. Zamani is with the Department of Computer Science, University of Colorado Boulder, USA. E-mail: majid.zamani@colorado.edu.

Note that all the aforementioned results on fault prognosis of DES consider dynamical systems with observations represented as event sequences which can always be precisely distinguished. However, in many practical CPS, observations are continuous real-valued signals, so that it may not be possible to confidently distinguish between two observations that are very close to each other due to imperfect measurement precision. To address this challenge, researchers have extended the notion of prognosability to metric systems in [5] by considering the proximity of observations and a finite-abstraction-based methodology was provided to verify approximate prognosability, where imperfect measurement precision is modeled by an accuracy parameter δ. However, this class of finite-abstraction-based approaches often suffer from the curse of dimensionality due to state-set discretizations, which can limit its scalability and applicability to larger and more complex systems.

In this paper, we investigate the verification of prognosability for discrete-time control systems with continuous state-space. Specifically, the notion of approximate (M, δ)-prognosability is considered, which requires that, under observation precision δ, every fault can be predicted before its occurrence, and once an alarm is issued, a fault will occur within M time instants from the alarm. Then, we introduce an abstraction-free approach to verify approximate (M, δ)-prognosability by reducing the verification problem into a safety verification problem and utilizing barrier certificates to solve it. Finally, we present a counterexample-guided inductive synthesis (CEGIS) framework to compute these certificates.

Recent literature has explored the verification of various approximate versions of information-flow properties in dynamical systems with imprecise observations [11]. For instance, in [19], the concept of approximate diagnosability was introduced for metric systems, ensuring that every occurrence of a fault can be detected within a finite time horizon. An abstraction-free approach for verifying approximate diagnosability was also proposed in [28]. In the context of information-flow security, the notion of approximate opacity was investigated in [12] and [27], with the development of verification algorithms based on barrier certificates in [9], [13] and [15]. Another related concept is approximate current-state observability, particularly in the context of state estimation under attacks [20]. However, as demonstrated in the discrete-event counterpart, the notion of prognosability is incomparable with above notions. In particular, prognosability deals with *future information*, presenting a specific challenge for the verification problem. To the best of our knowledge, there has been no exploration of abstraction-free

methodologies for verifying approximate prognosability in the literature.

II. Preliminaries

A. Notations

Let $\mathbb{R}$ and $\mathbb{N}$ be the set of real numbers and non-negative integers, respectively. The set of non-negative real numbers is denoted by $\mathbb{R}_{\geq 0}$. For $a, b \in \mathbb{N}$ and $a \leq b$, the closed interval in $\mathbb{N}$ is denoted by $[a, b]$. Consider a set X, a string $s = (x_0, x_1, \cdots, x_n(\cdots))$ is a finite (or infinite) sequence over X if $x_i \in X$ for all $i = 0, 1, \cdots, n(\cdots)$. We denote by X^* and X^ω the set of all finite and infinite strings over X, respectively, and use $X^+ = X^* \cup X^\omega$ to denote the set of all finite or infinite strings. For a finite string $s \in X^*$, we denote by $|s|$ the number of components in s. For a string $s = (x_0, x_1, \cdots) \in X^+$ and non-negative integers $i, j \in \mathbb{N}, i \leq j$, we denote by $s(i) = x_i$ its i^{th} item and by $s[i\!:\!j] = (x_i, \cdots, x_j)$ the segment between the i^{th} items and j^{th} items in the string s. Given a vector $x \in \mathbb{R}^n$, we denote by $\|x\|$ the Euclidean norm of x. For any two sets X and Y, we define the complement of X with respect to Y as $Y \backslash X = \{x \in Y : x \notin X\}$. For a set $X \subseteq \mathbb{R}^n$, the boundary and topological closure of X are denoted by ∂X and $\mathsf{clo}(X)$, respectively.

B. System Model

In this work, we consider a discrete-time control system (dt-CS) G by the tuple:

$$G := (X, X_0, U, f, Y, h),$$

where $X \subseteq \mathbb{R}^n, U \subseteq \mathbb{R}^m$ and $Y \subseteq \mathbb{R}^p$ are the set of states, inputs and outputs, respectively; $X_0 \subseteq X$ is the set of initial state; the function $f : X \times U \to X$ is the transition function and $h : X \to Y$ is the output map. A dt-CS can also be described by the following difference equations:

$$G : \begin{cases} \mathbf{x}(t+1) = f(\mathbf{x}(t), \mathbf{u}(t)), \\ \mathbf{y}(t) = h(\mathbf{x}(t)), \end{cases} \tag{1}$$

where $t \in \mathbb{N}$ and $\mathbf{x} : \mathbb{N} \to X$, $\mathbf{u} : \mathbb{N} \to U$, $\mathbf{y} : \mathbb{N} \to Y$ are state, input, and output strings. We denote by $\mathcal{U}$ the set of all input strings. Given an input string $\mathbf{u} \in \mathcal{U}$ and an initial state $x_0 \in X_0$, the state string starting from x_0 under input $\mathbf{u}$ is denoted by $\mathbf{x}_{x_0,\mathbf{u}} = (x_0, x_1, \dots)$ such that for any $t \in \mathbb{N}$, $x_{t+1} = f(x_t, \mathbf{u}(t))$. Then, the set of state strings generated by G is denoted by $\mathsf{Path}(G) = \{\mathbf{x}_{x_0,\mathbf{u}} = (x_0, x_1, \dots) \in X^+ : \exists x_0 \in X_0, \exists \mathbf{u} \in \mathcal{U}, \forall t \in \mathbb{N}, x_{t+1} = f(x_t, \mathbf{u}(t))\}$. The output map h is extended to $h : X^+ \to Y^+$ such that for any state string $\mathbf{x}_{x_0,\mathbf{u}} = (x_0, x_1, \dots) \in \mathsf{Path}(G)$, we have $h(\mathbf{x}_{x_0,\mathbf{u}}) = (h(x_0), h(x_1), \dots)$. For the sake of simplicity, we also denote by $\mathbf{y}_{x_0,\mathbf{u}} = h(\mathbf{x}_{x_0,\mathbf{u}})$ the output of $\mathbf{x}_{x_0,\mathbf{u}}$.

To model the imprecise observation in real world applications, here, we introduce the notion of *observation precision*, denoted by $\delta \in \mathbb{R}_{\geq 0}$, to model the observation error. Furthermore, given an observation precision δ, we say two state $x, \hat{x} \in X$ are *output equivalent* if $\|h(x) - h(\hat{x})\| \leq \delta$, which is denoted by $h(x) \approx h(\hat{x})$. Similarly, we say two state strings $\mathbf{x}_{x_0,\mathbf{u}}, \mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}} \in \mathsf{Path}(G)$ are output equivalent if $\|\mathbf{y}_{x_0,\mathbf{u}}(t)) - \mathbf{y}_{\hat{x}_0,\hat{\mathbf{u}}}(t)\| \leq \delta$ for all $t \in \mathbb{N}$, which, with a slight abuse of notation, is denoted by $\mathbf{y}_{x_0,\mathbf{u}} \approx \mathbf{y}_{\hat{x}_0,\hat{\mathbf{u}}}$. Then, given a state string $\mathbf{x}_{x_0,\mathbf{u}} \in X^+$, we define the set of all *possible observations* of $\mathbf{x}_{x_0,\mathbf{u}}$ by $H(\mathbf{x}_{x_0,\mathbf{u}}) := \{\mathbf{y}_{\hat{x}_0,\hat{\mathbf{u}}} \in Y^+ : \mathbf{y}_{x_0,\mathbf{u}} \approx \mathbf{y}_{\hat{x}_0,\hat{\mathbf{u}}}\}$.

III. Approximate (M, δ)-Prognosability for Discrete-Time Control Systems

In the context of fault prognosis problem, the goal is to alarm whether or not the failures will occur. To this end, we define by $X_F \in X$ the faulty state set. We say a state string $\mathbf{x}_{x_0,\mathbf{u}} \in \mathsf{Path}$ is faulty if it enters the faulty state set, i.e., $\mathbf{x}_{x_0,\mathbf{u}}(k) \in X_F$ for some $k \in \mathbb{N}$. Note that we do not consider the case of fault recovery in this work, that is, once the system enters the faulty state set, it remains faulty indefinitely. We denote by $\Psi(G)$ the set of finite state strings in which faulty states appear *for the first time*, i.e.,

$$\begin{aligned}\Psi(G) :=& \{\mathbf{x} = (x_0, \dots, x_k) \in \mathsf{Path}(G) : x_k \in X_F \\ & \wedge (\forall t \in [0, k-1])(x_t \notin X_F)\}.\end{aligned} \tag{2}$$

Then, we formally define approximate (M, δ)-prognosability as follows:

Definition 1 (Approximate (M, δ)-prognosability): Consider $M \in \mathbb{N}$ and $\delta \in \mathbb{R}_{\geq 0}$. A dt-CS $G = (X, X_0, U, f, Y, h)$ with observation precision δ is said to be approximate (M, δ)-prognosable if and only if

$$\begin{aligned}&(\forall \mathbf{x}_{x_0,\mathbf{u}} = (x_0, \dots, x_k) \in \Psi(G))(\exists t \in [0, k-1])) : \\ &(\forall \mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}} \in \mathsf{Path}(G))((\mathbf{y}_{\hat{x}_0,\hat{\mathbf{u}}}[0\!:\!t] \in H(\mathbf{x}_{x_0,\mathbf{u}}[0\!:\!t]) \\ &\wedge (\forall i \in [0, t])(\mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}}(i) \notin X_F)) \Rightarrow \\ &((\exists j \in [t+1, t+M])(\mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}}(j) \in X_F))).\end{aligned}$$

Intuitively, the above definition says that, for any faulty state string $\mathbf{x}_{x_0,\mathbf{u}} = (x_0, \dots, x_k) \in \Psi(G)$, it must have a normal prefix such that the prognoser can predict for sure that fault must occur in the next M steps based on its observation. The above notion modifies the approximate prognosability given in [5] by utilizing a non-negative integer M to flexibly quantify the required time bound within which a fault is guaranteed to occur from the time the prognosis alarm has been issued.

In order to be able to prognose failures of a dt-CS in real time, we aim to construct a prognoser such that it can always predict the faults before their occurrence and fault will occur for sure within the next M steps once a prognosis alarm is issued, which is referred as a (M, δ)-*prognoser*. Formally speaking, a (M, δ)-prognoser is defined as a function $D : Y^* \to \{0, 1\}$, which provides "1" if a failure is inevitable within the next M steps, and provides "0", otherwise. We say that a (M, δ)-prognoser works correctly if it satisfies the following conditions:

C1) Any failure can be predicted before its occurrence, i.e.,

$$\begin{aligned}&(\forall \mathbf{x}_{x_0,\mathbf{u}} = (x_0, \dots, x_k) \in \Psi(G))(\exists t \in [0, k-1]) : \\ &(\forall \mathbf{y} \in H(\mathbf{x}_{x_0,\mathbf{u}}))(\forall t' \in [t, k-1])(D(\mathbf{y}[0 : t']) = 1),\end{aligned}$$

C2) Once a prognose alarm is issued, failure is guaranteed to occur within next M steps, i.e.,

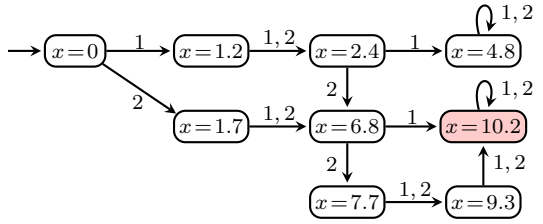

Fig. 1: A finite dt-CS for approximate (M,δ)-prognosability as a running example, where states marked by red denote faulty state.

$$(\forall \mathbf{x}_{x_0,\mathbf{u}} \in \mathsf{Path}(G))(\forall t \in \mathbb{N}) : [((\exists \mathbf{y} \in H(\mathbf{x}_{x_0,\mathbf{u}}))$$
$$(D(\mathbf{y}[0\!:\!t])\!=\!1)) \Rightarrow$$
$$(\exists t' \in [t+1, t+M])(\mathbf{x}_{x_0,\mathbf{u}}(t') \in X_F)].$$

The following theorem shows that approximate (M,δ)-prognosability indeed provides the necessary and sufficient condition for the existence of an (M,δ)-prognoser.

Theorem 1: Consider a dt-CS G. There exists a (M,δ)-prognoser satisfying conditions C1 and C2 if and only if the dt-CS G is approximate (M,δ)-prognosable.

To better illustrate our previous results, we consider the following dt-CS with finite state as our running example.

Example 1: Let us consider a dt-CS $G = (X, X_0, U, f, Y, h)$ depicted in Figure 1, where $X = \{0, 1.2, 1.7, 2.4, 4.8, 6.8, 7.7, 9.3, 10.2\}, X_0 = \{0\}, X_F = \{10.2\}, U = \{1,2\}, Y = X$, and the output map $h : X \to Y$ is defined as $h(x) := x$ for any $x \in X$. We claim that G is not approximate $(2,1)$-prognosable. For example, let's consider input $\mathbf{u} = (2,2,1)$ and $x_0 = 0$. We have a finite state string $\mathbf{x}_{x_0,\mathbf{u}} = (0, 1.7, 6.8, 10.2) \in \Psi(G)$. There exists $\hat{\mathbf{u}} = (2,2,2,2)$ and $\hat{x}_0 = 0$ such that $\mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}} = (0, 1.7, 6.8, 7.7, 9.3) \in \mathsf{Path}(G)$ and $h(\mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}}[0:2]) \approx h(\mathbf{x}_{x_0,\mathbf{u}}[0:2])$. Since $\mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}}(i) \notin X_F$ for all $i \in [0,4]$, by Definition 1, system G is not approximate $(2,1)$-prognosable. On the other hand, one can check that the system is approximate $(3,1)$-prognosable.

IV. Verification of approximate (M,δ)-prognosability

In this section, we provide a necessary and sufficient condition under which a dt-CS is approximate (M,δ)-prognosable based on the verification system.

A. Augmented Systems

Given a dt-CS $G = (X, X_0, U, f, Y, h)$, the *augmented system* associated with G is defined as

$$\bar{G} := (\bar{X}, \bar{X}_0, \bar{U}, \bar{f}, \bar{Y}),$$

where $\bar{X} = X \times X, \bar{X}_0 = X_0 \times X_0, \bar{U} = U \times U, \bar{Y} = Y \times Y$ and $\bar{f} : \bar{X} \times \bar{U} \to \bar{X}$ is the transition function defined by: for any $\bar{x} = (x, \hat{x}) \in \bar{X}$ and $\bar{u} = (u, \hat{u}) \in \bar{U}$, we have $\bar{f}(\bar{x}, \bar{u}) = (f(x,u), f(\hat{x}, \hat{u}))$. Essentially, $\bar{G}$ is constructed by augmenting dt-CS G with itself. Given an initial state $\bar{x}_0 = (x_0, \hat{x}_0) \in \bar{X}_0$ and an input string $\bar{\mathbf{u}} = (\mathbf{u}, \hat{\mathbf{u}}) \in \mathcal{U} \times \mathcal{U}$, we have the state string $\bar{\mathbf{x}}_{\bar{x}_0,\bar{\mathbf{u}}} = ((x_0, \hat{x}_0), (x_1, \hat{x}_1), \dots)$ in $\bar{G}$ starting from $\bar{x}_0$ under input $\bar{\mathbf{u}}$ such that $\mathbf{x}_{x_0,\mathbf{u}} = (x_0, x_1, \dots) \in \mathsf{Path}(G)$ and $\mathbf{x}_{\hat{x}_0,\hat{\mathbf{u}}} = (\hat{x}_0, \hat{x}_1, \dots) \in \mathsf{Path}(G)$. We denote by $\mathsf{Path}(\bar{G})$ the set of state string generated by $\bar{G}$.

B. Verification Structure

Given an augmented system $\bar{G}$, we first define three state partitions in $\bar{G}$: (i) both components are output equivalent, $\Upsilon_1 = \{(x, \hat{x}) \in \bar{X} : h(x) \approx h(\hat{x})\}$; (ii) first component is faulty, $\Upsilon_2 = \{(x, \hat{x}) \in \bar{X} : x \in X_F\}$; and (iii) second component is faulty, $\Upsilon_3 = \{(x, \hat{x}) \in \bar{X} : \hat{x} \in X_F\}$. Based on above state partitions, we define a set of event $\Sigma = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6\}$ and a labeling function $L : \bar{X} \to 2^{\Sigma}$ such that $L^{-1}(\sigma_1) = \Upsilon_1 \cap (\bar{X} \backslash \Upsilon_2) \cap (\bar{X} \backslash \Upsilon_3)$, $L^{-1}(\sigma_2) = \Upsilon_2 \cap (\bar{X} \backslash \Upsilon_3)$, $L^{-1}(\sigma_3) = \bar{X} \backslash \Upsilon_3$, $L^{-1}(\sigma_4) = \Upsilon_3$, $L^{-1}(\sigma_5) = (\bar{X} \backslash \Upsilon_1) \cup \Upsilon_3$, and $L^{-1}(\sigma_6) = \bar{X}$. Intuitively, for a state $\bar{x} = (x, \hat{x}) \in \bar{X}$, $\sigma_1 \in L(\bar{x})$ denotes both component of $\bar{x}$ are output equivalent and normal; $\sigma_2 \in L(\bar{x})$ denotes that x is faulty but $\hat{x}$ is normal; $\sigma_3 \in L(\bar{x})$ and $\sigma_4 \in L(\bar{x})$ denote $\hat{x}$ is normal and faulty, respectively; $\sigma_5 \in L(\bar{x})$ denotes x and $\hat{x}$ are not output equivalent or $\hat{x}$ is faulty, and $\sigma_6 \in L(\bar{x})$ is held for any $\bar{x} \in \bar{X}$.

After assigning different labels to states of augment system $\bar{G}$, we devise M-deterministic finite automaton (M-DFA) based on event set Σ to record the label sequences of state paths of $\bar{G}$ as follows.

Definition 2 (M-deterministic finite automaton): Consider $M \in \mathbb{N}$. An M-DFA is a four-tuple: $A := (Q, q_0, \Sigma, E)$, where $Q = \{q_0, q_1, \dots, q_M, q_t\}$ is the finite set of states, $q_0 \in Q$ is the initial state, $\Sigma = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6\}$ is the finite set of events, $E : Q \times \Sigma \to Q$ is the transition function such that: $q_0 = E(q_0, \sigma_1)$; $q_1 = E(q_0, \sigma_2)$; $q_t = E(q_0, \sigma_5)$; $q_M = E(q_M, \sigma_6)$; for any $i = [1, M-1]$, $q_{i+1} = E(q_i, \sigma_3)$ and $q_t = E(q_i, \sigma_4)$.

Given a non-negative integer M, the template of M-DFA is depicted in Figure 2. Intuitively, M-DFA is a DFA with $M+2$ states and driven by labels of states in augmented system $\bar{G}$. Considering a finite state string $\bar{\mathbf{x}}_{\bar{x}_0,\bar{\mathbf{u}}} = ((x_0, \hat{x}_0), \dots, (x_n, \hat{x}_n)) \in \mathsf{Path}(\bar{G})$, M-DFA stays at state q_0 after monitoring $\bar{\mathbf{x}}_{\bar{x}_0,\bar{\mathbf{u}}}$ if only event σ_1 occurs, which means $(x_0, \dots, x_n)$ and $(\hat{x}_0, \dots, \hat{x}_n)$ are normal and output equivalent. If M-DFA moves to state q_1 driven by $\bar{\mathbf{x}}_{\bar{x}_0,\bar{\mathbf{u}}}$, we know that $(x_0, \dots, x_n)$ enters into fault region for the first time, $(\hat{x}_0, \dots, \hat{x}_n)$ is still normal and both of them are output equivalent before the previous step, i.e., $(x_0, \dots, x_n) \in \Psi(G), h((x_0, \dots, x_{n-1})) \approx h((\hat{x}_0, \dots, \hat{x}_{n-1}))$ and $\hat{x}_i \notin X_F$ for all $i \in [0,n]$. Suppose M-DFA reaches state $q_k, k \in [2, M]$, after monitoring $\bar{\mathbf{x}}_{\bar{x}_0,\bar{\mathbf{u}}}$, $(x_k, \hat{x}_k), k \in [0,n]$. By the definition of event σ_3, we know that $x_i \in X_F$ for some $i \in [0,n]$ and $\hat{x}_j \notin X_F$ for all $j \in [0,n]$. Note that according to the definition of labels σ_3 and σ_2, we do not require state x and $\hat{x}$ to be observable equivalent for all $\bar{x} = (x, \bar{x}) \in \bar{X}$ when $\sigma_3 \in L(\bar{x})$ or $\sigma_2 \in L(\bar{x})$. Specifically, after M-DFA leaving q_0, we relax the constraint of observation equivalence.

Now, based on the augmented system $\bar{G}$ and M-DFA, we construct the verification system as follows:

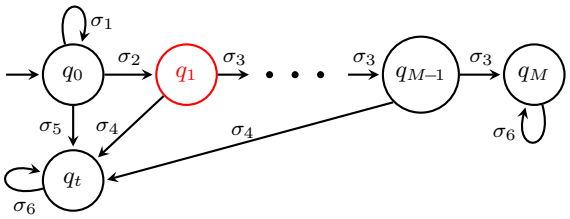

Fig. 2: The template of M-DFA

.

Definition 3 (Verification system): Given a dt-CS G, its verification system is a tuple $V := (\tilde{X}, \tilde{X}_0, \tilde{U}, \tilde{f}, \mathcal{F})$, where $\tilde{X} = X \times X \times Q$, $\tilde{X}_0 = \{(x_0, \hat{x}_0, q) \in X_0 \times X_0 \times Q : \exists \sigma \in L(x_0, \hat{x}_0), q = E(q_0, \sigma)\}$, $\tilde{U} = U \times U$, and $\tilde{f} : \tilde{X} \times \tilde{U} \to \tilde{X}$ is a deterministic transition function defined by: for any $\tilde{x} = (x, \hat{x}, q) \in \tilde{X}$ and $\tilde{u} = (u, \hat{u}) \in \tilde{U}$, we have $\tilde{f}(\tilde{x}, \tilde{u}) = (x', \hat{x}', q')$ if $x' = f(x, u), \hat{x}' = f(\hat{x}, \hat{u})$ and there exists $\sigma \in L(x', \hat{x}')$ such that $q' = E(q, \sigma)$, and $\mathcal{F} = \bar{X} \times \{q_M\}$ is the set of accepting states.

Essentially, the verification system V is constructed by synchronizing the augmented system $\bar{G}$ with the M-DFA according to the labeling function $L : \bar{X} \to 2^{\Sigma}$. Specifically, for any state $\tilde{x} = (x, \hat{x}, q) \in \tilde{X}$ and input $\tilde{u} = (u, \hat{u}) \in \tilde{U}$, one gets $f(\tilde{x}, \tilde{u}) = (x', \hat{x}', q')$ whenever (i) $x' = f(x, u), \hat{x}' = f(\hat{x}, \hat{u})$; and (ii) there exists a label of the state pair $(x', \hat{x}')$ satisfying the transition from q to q', i.e., $\exists \sigma \in L(x, \hat{x}), q' = E(q, \sigma)$. We denote the set of all input string pairs by $\mathcal{U}_V = \mathcal{U} \times \mathcal{U}$. Then, given an input string $\tilde{\mathbf{u}} \in \mathcal{U}_V$ and an initial state $\tilde{x}_0 \in \tilde{X}_0$, we denote by $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}} = (\tilde{x}_0, \tilde{x}_1, \dots)$ the state string in the verification system V from initial state $\tilde{x}_0$ under the input $\tilde{\mathbf{u}}$. The set of state strings generated by V is denoted by $\mathsf{Path}(V)$. For state $\tilde{x} = (x, \hat{x}, q) \in \tilde{X}$, we denote by $\theta_1(\tilde{x}) = x, \theta_2(\tilde{x}) = \hat{x}$ and $\theta_3(\tilde{x}) = q$ its first, second, and third component, respectively. We extend this notation to state string $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}} = (\tilde{x}_0, \tilde{x}_1, \dots) \in \mathsf{Path}(V)$ by $\theta_i(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}) = (\theta_i(\tilde{x}_0), \theta_i(\tilde{x}_1), \dots)$ for $i \in \{1, 2, 3\}$. By construction, V has the following two properties:

- For any state string $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}} = (\tilde{x}_0, \tilde{x}_1, \dots, \tilde{x}_n) \in \mathsf{Path}(V)$ such that $\theta_3(\tilde{x}_n) = q_1, \tilde{x}_0 = (x_0, \hat{x}_0)$ and $\tilde{\mathbf{u}} = (\mathbf{u}, \hat{\mathbf{u}})$, we have two state strings $\mathbf{x}_{x_0, \mathbf{u}} = \theta_1(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}), \mathbf{x}_{\hat{x}_0, \hat{\mathbf{u}}} = \theta_2(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}) \in \mathsf{Path}(G)$ such that $\mathbf{y}_{x_0, \mathbf{u}}[0 : n-1] \approx \mathbf{y}_{\hat{x}_0, \hat{\mathbf{u}}}[0 : n-1], \mathbf{x}_{x_0, \mathbf{u}} \in \Psi(G)$ and for any $i \in [0, n], \mathbf{x}_{\hat{x}_0, \hat{\mathbf{u}}}(i) \notin X_F$;
- For any pair of state strings $\mathbf{x}_{x_0, \mathbf{u}}, \mathbf{x}_{\hat{x}_0, \hat{\mathbf{u}}} \in \mathsf{Path}(G)$, there exists $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}} \in \mathsf{Path}(V)$ such that $\theta_1(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}) = \mathbf{x}_{x_0, \mathbf{u}}, \theta_2(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}) = \mathbf{x}_{\hat{x}_0, \hat{\mathbf{u}}}, \tilde{x}_0 = (x_0, \hat{x}_0)$, and $\tilde{\mathbf{u}} = (\mathbf{u}, \hat{\mathbf{u}})$.

We are now ready to recast the verification of approximate (M, δ)-prognosability as a safety verification problem for the verification system.

Theorem 2: Consider a dt-CS G. The dt-CS G is not approximate (M, δ)-prognosable if and only if in the verification system V, there exists $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}} \in \mathsf{Path}(V)$ such that $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}(k) \in \mathcal{F}$ for some $k \in \mathbb{N}$.

Lets intuitively explain the above theorem. Without loss of generality, suppose there exists $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}} \in \mathsf{Path}(V)$ that reaches a state in $\mathcal{F}$ for the first time at time step k, i.e., for some $k \in \mathbb{N}$, $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}(k) \in \mathcal{F}$ and for any $l \in [0, k-1]$, $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}(l) \notin \mathcal{F}$. By the construction of M-DFA, let $i = k - M + 1$. We have $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}(i) \in \bar{X} \times \{q_1\}$. Let $\mathbf{x}_{x_0, \mathbf{u}} = \theta_1(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}})$ and $\mathbf{x}_{\hat{x}_0, \hat{\mathbf{u}}} = \theta_2(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}})$. According to the first property of verification system V, we know that the prefix $\mathbf{x}_{x_0, \mathbf{u}}[0 : i]$ of $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}$ is included in $\Psi(G)$, i.e., $\mathbf{x}_{x_0, \mathbf{u}}[0 : i] \in \Psi(G)$. Furthermore, let $t \in \max\{0, i-1\}$. Then one has $\mathbf{y}_{\hat{x}_0, \hat{\mathbf{u}}}[0 : t] \approx \mathbf{y}_{x_0, \mathbf{u}}[0 : t]$ and $\mathbf{x}_{\hat{x}_0, \hat{\mathbf{u}}}(j) \notin X_F$ for all $j \in [0, t]$. Since $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}$ reaches a state in $\mathcal{F}$ at the k^{th} time step, i.e., $\theta_3(\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}(k)) = q_M$, by the definition of events σ_2 and σ_3, we know that $\mathbf{x}_{\hat{x}_0, \hat{\mathbf{u}}}(m) \notin X_F$ for all $m \in [t+1, t+M]$. Therefore, the existence of such $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}} \in \mathsf{Path}(V)$ satisfying $\tilde{\mathbf{x}}_{\tilde{x}_0, \tilde{\mathbf{u}}}(k) \in \mathcal{F}$ for some $k \in \mathbb{N}$ falsifies the desired approximate (M, δ)-prognosability property.

V. Barrier Certificates for Checking (Lack of) Approximate (M, δ)-Prognosability

In this section, we provide barrier certificates to verify (lack of) approximate (M, δ)-prognosability.

A. Verifying Approximate (M, δ)-Prognosability via Barrier Certificates

In this section, we leverage a notion of *barrier certificate* to check (M, δ)-prognosability. The following theorem provides a sufficient condition for approximate (M, δ)-prognosability.

Theorem 3: Consider a dt-CS G. The dt-CS G is approximate (M, δ)-prognosable w.r.t observation precision δ and constant M if for its verification system $V = (\tilde{X}, \tilde{X}_0, \tilde{U}, \tilde{f}, \mathcal{F})$, there exists a function $\mathcal{B} : \tilde{X} \to \mathbb{R}$ such that

$$\forall \tilde{x} \in \tilde{X}_0, \qquad \mathcal{B}(\tilde{x}) \leq 0, \tag{3}$$

$$\forall \tilde{x} \in \mathcal{F}, \qquad \mathcal{B}(\tilde{x}) > 0, \tag{4}$$

$$\forall \tilde{x} \in \bar{X} \times Q \setminus \{q_t\}, \forall \tilde{u} \in \tilde{U}, \qquad \mathcal{B}(\tilde{f}(\tilde{x}, \tilde{u})) \leq \mathcal{B}(\tilde{x}). \tag{5}$$

Since the state set Q of the M-DFA is finite and discrete, the state set $\tilde{X}$ is a hybrid state set. To compute $\mathcal{B}$, we first define a function $\mathsf{Ind} : Q \to [0, M] \cup \{t\}$ that maps the state $q_i \in Q$ to its the index i. Then, we fix the template of barrier certificates as: for any $\tilde{x} = (x, \hat{x}, q) \in \tilde{X}$,

$$\mathcal{B}(\tilde{x}) = \begin{cases} \mathcal{B}_l(x, \hat{x}) = \sum_{n=1}^{z_l^b} \alpha_l^n p_l^n(x, \hat{x}), \text{if } l = \mathsf{Ind}(q) \in [0, M], \\ \mathcal{B}_t(x, \hat{x}) = \sum_{n=1}^{z_t^b} \alpha_t^n p_t^n(x, \hat{x}), \text{if } \mathsf{Ind}(q) = t, \end{cases} \tag{6}$$

where for any $l \in [0, M]$, $p_l^n(x, \hat{x})$ and $p_t^n(x, \hat{x})$ are monomials over the state variables x and $\hat{x}$ with fixed degree $d_b \in \mathbb{N}$, α_l^n and α_t^n are unknown coefficients, $z_t^b, z_l^b \in \mathbb{N}$ are some positive integers.

We use the counterexample-guided inductive synthesis (CEGIS) framework [8], [22] to find barrier certificates. Given a state $q \in Q$, the state which M-DFA will reach from state $q \in Q$ after monitoring a state pair $\bar{x} = (x, \hat{x}) \in \bar{X}$ is given by: $\Theta(q, \bar{x}) = q'$, if there exists $\sigma \in L(\bar{x})$ such that $q' = E(q, \sigma)$. Given states $q, q' \in Q$, the set of all state pairs and input pairs $(x, \hat{x}, u, \hat{u}) \in \bar{X} \times \bar{U}$, such that M-DFA will

reach state q' from q after monitoring $(f(x,u), f(\hat{x},\hat{u}))$ is given by

$$\mathsf{Nex}(q,q') = \{(x,\hat{x},u,\hat{u}) \in \bar{X}\times\bar{U} : \exists\sigma \in L(f(x,u), f(\hat{x},\hat{u})), q' = E(q,\sigma)\}. \quad (7)$$

Then, based on above operators, we encode the definition of barrier certificates in inequations (3)-(5) as the following conjunction of inequations:

$$\Phi_1^b := \bigwedge_{\bar{x}\in\bar{X}_0, q\in\Theta(q,\bar{x})} \mathcal{B}_{\text{Ind}(q)}(\bar{x}) \le 0; \quad (8)$$

$$\Phi_2^b := \bigwedge_{\bar{x}\in\bar{X}} \mathcal{B}_M(\bar{x}) > 0; \quad (9)$$

$$\Phi_3^b := \bigwedge_{\substack{q\in Q\setminus\{q_t\}, q'\in\Gamma(q),\\ (x,\hat{x},u,\hat{u})\in\mathsf{Nex}(q,q')}} \mathcal{B}_{\text{Ind}(q')}(f(x,u),f(\hat{x},\hat{u}))) \le \mathcal{B}_{\text{Ind}(q)}(x,\hat{x}). \quad (10)$$

Given the barrier certificate template as in Equation (6), the CEGIS framework for computing $\mathcal{B}$ is as follows:

1) We select finite set of samples $\bar{X}_B \subset \bar{X}$ and $\bar{U}_B \subset \bar{U}$, respectively.
2) Compute a candidate $\mathcal{B}$ with template defined in Equation (6) such that for any state $\bar{x} \in \bar{X}_B$ and input $\bar{u} \in \bar{U}_B$, the formula $\Phi^b := \Phi_1^b \wedge \Phi_2^b \wedge \Phi_3^b$ is true. By substituting each $\bar{x} \in \bar{X}_B$ and $\bar{u} \in \bar{U}_B$ into Φ^b, the above computation of candidate $\mathcal{B}$ is reduced to a linear programming problem with decision variables $\alpha_l^n, l \in [0,M]$ and α_t^n, which can be solved by the off-the-shelf solvers, such as CVXPY [6]. If such candidate $\mathcal{B}$ does not exist, we conclude that there is no barrier certificate $\mathcal{B}$ with template defined in Equation (6).
3) If there is a feasible solution for candidate $\mathcal{B}$, in order to check whether it is indeed a true barrier certificate satisfying Φ^b for all $\bar{x} \in \bar{X}$ and $\bar{u} \in \bar{U}$, we search for some counterexamples $\bar{x}_c \in \bar{X}$ and $\bar{u}_c \in \bar{U}$ such that $\neg\Phi^b$ is true. We can encode $\neg\Phi^b$ as a Satisfiability Modulo Theories (SMT) query and solve it by SMT solver, such as Z3 [4].
4) If the counterexample cannot be found, i.e., $\neg\Phi^b$ has no feasible solution, the candidate $\mathcal{B}$ is a valid barrier certificate. Otherwise, if $\neg\Phi^b$ has feasible solution $\bar{x}_c$ and $\bar{u}_c$, we obtain new sample set by adding the counterexamples: $\bar{X}_B \cup \{\bar{x}_c\}$ and $\bar{U}_B \cup \{\bar{u}_c\}$, and repeat Steps 2)-4).

Example 2: Let us consider the dt-CS G shown in Figure 1 with observation precision $\delta = 1$. Let $M = 3$. To check if system G is approximate $(3,1)$-prognosable, we utilize the CEGIS framework and compute a barrier certificate $\mathcal{B}$ satisfying Equation (8)-(10) as shown in the following equation under the template in Equation (6) with degree 3.

$$\mathcal{B}(\tilde{x}) = \begin{cases} \mathcal{B}_0(x,\hat{x}) = -0.6055 - 0.3879x - 0.3879\hat{x} + \\ 0.1548x^2 - 0.0640x\hat{x} + 0.1548\hat{x}^2 - 0.1318x^3 + \\ 0.1036x^2\hat{x} + 0.1036x\hat{x}^2 - 0.1318\hat{x}^3, \text{ if } q = q_0; \\ \mathcal{B}_1(x,\hat{x}) = 0.7116 - 0.5457x - 0.0956x^2 - \\ 0.0168x^3, \qquad \text{if } q = q_1; \\ \mathcal{B}_2(x,\hat{x}) = 2.3439 - 0.7707x - 0.7707\hat{x} - \\ 0.7282x^2 + 0.2826x\hat{x} - 0.7282\hat{x}^2 - 0.4690x^3 + \\ 0.6892x^2\hat{x} + 0.6892x\hat{x}^2 - 0.4690\hat{x}^3, \text{ if } q = q_2; \\ \mathcal{B}_3(x,\hat{x}) = 1.8874 - 0.6633x - 0.6633\hat{x} + \\ 0.1503x^2 + 0.2311x\hat{x} + 0.1503\hat{x}^2 + 0.0235x^3 - \\ 0.0280x^2\hat{x} - 0.0280x\hat{x}^2 + 0.0235\hat{x}^3, \text{ if } q = q_3; \\ \mathcal{B}_t(x,\hat{x}) = -1.3440 + 0.2775x + 0.2477\hat{x} - \\ 0.0944x^2 - 0.1661x\hat{x} - 0.0795\hat{x}^2 - 0.0249x^3 + \\ 0.0255x^2\hat{x} + 0.0195x\hat{x}^2 - 0.0256\hat{x}^3, \text{ if } q = q_{\text{t}}. \end{cases}$$

Therefore, system G is approximate $(3,1)$-prognosable according to Theorem 3.

B. Verifying Lack of Approximate (M,δ)-prognosability via Barrier Certificates

In this section, we propose another type of *barrier certificate* $\mathcal{V} : \tilde{X} \to \mathbb{R}$ for verification system V to check the lack of approximate (M,δ)-prognosability.

Theorem 4: Consider a dt-CS G. The dt-CS G is not approximate (M,δ)-prognosable w.r.t observation precision δ and constant M if for its verification system $V = (\tilde{X}, \tilde{X}_0, \tilde{U}, \tilde{f}, \mathcal{F})$, there exists a function $\mathcal{V} : \tilde{X} \to \mathbb{R}$ such that

$$\forall\tilde{x} \in \tilde{X}_0, \qquad \mathcal{V}(\tilde{x}) \le 0; \quad (11)$$

$$\forall\tilde{x} \in \partial\bar{X} \times (Q \setminus \{q_M\}), \qquad \mathcal{V}(\tilde{x}) > 0; \quad (12)$$

$$\forall\tilde{x} \in \mathsf{clo}(\tilde{X} \setminus \mathcal{F}), \exists\tilde{u} \in \tilde{U}, \quad \mathcal{V}(\tilde{f}(\tilde{x},\tilde{u})) < \mathcal{V}(\tilde{x}). \quad (13)$$

Similarly, we consider a template for barrier certificate $\mathcal{V}$ to based on different states of M-DFA as follows:

$$\mathcal{V}(\tilde{x}) = \begin{cases} \mathcal{V}_l(x,\hat{x}) = \sum_{n=1}^{z_l^v} \beta_l^n w_l^n(x,\hat{x}), \text{if } l = \mathsf{Ind}(q) \in [0,M], \\ \mathcal{V}_t(x,\hat{x}) = \sum_{n=1}^{z_t^v} \beta_t^n w_t^n(x,\hat{x}), \text{if } \mathsf{Ind}(q) = t, \end{cases} \quad (14)$$

where for any $l \in [0,M]$, $w_l^n(x,\hat{x})$ and $w_t^n(x,\hat{x})$ are monomials over the state variables x and $\hat{x}$ with fixed degree $d_v \in \mathbb{N}$, β_l^n and β_t^n are unknown coefficients, $z_t^v, z_l^v \in \mathbb{N}$ are some positive integers.

To compute barrier certificate $\mathcal{V}$ via the CEGIS framework, we formulate the conditions of $\mathcal{V}$ in Theorem 4 as the following inequations:

$$\Phi_1^v = \bigwedge_{\bar{x}\in\bar{X}_0, q\in\Theta(q,\bar{x})} \mathcal{V}_{\text{Ind}(q)}(\bar{x}) \le 0; \quad (15)$$

$$\Phi_2^v = \bigwedge_{\bar{x}\in\partial\bar{X}, q\in(Q\setminus\{q_M\})} \mathcal{V}_{\text{Ind}(q)}(\bar{x}) > 0; \quad (16)$$

$$\Phi_3^v = \bigwedge_{\substack{\bar{x}\in\mathsf{clo}(\bar{X}),\\ q\in Q\setminus\{q_M\}}} \bigvee_{\substack{\bar{u}\in\bar{U},\\ q'\in\Theta(q,\bar{f}(\bar{x},\bar{u}))}} \mathcal{V}_{\text{Ind}(q')}(\bar{f}(\bar{x},\bar{u}))) \le \mathcal{V}_{\text{Ind}(q)}(\bar{x}), \quad (17)$$

Given the barrier certificate template in Equation (14), we can also use the similar CEGIS framework for computing $\mathcal{B}$ to compute $\mathcal{B}$.

Remark 1: If a dt-CS G has a polynomial transition function and its output map h is a polynomial function in variable x, one can alternatively make use of sum-of-squares (SOS) programming to search for barrier certificates as in [21], with the help of semidefinite programming tools, such as SOSTOOLS [17] and SeDuMi [23]. Recently, machine learning techniques also have been exploited to search for barrier certificates and the readers are referred to [18] for more details.

Example 3: We again still consider the dt-CS G depicted in Figure 1 with observation precision $\delta = 1$. In this case, we consider $M = 2$. To check if system G is approximate $(2, 1)$-prognosable, we utilize the CEGIS framework to compute a barrier certificate satisfying inequations (15)-(17) as shown in the following equation under the template in Equation (6) with degree 3.

$$\mathcal{V}(\tilde{x})=\begin{cases}\mathcal{V}_0(x,\hat{x}) = -0.0267 - 0.6546x - 0.6546\hat{x}+\\ 8.1664x^2 - 17.0279x\hat{x} + 8.1664\hat{x}^2 - 0.7584x^3+\\ 0.8023x^2\hat{x} + 0.8023x\hat{x}^2 - 0.7584\hat{x}^3, \text{ if } q = q_0;\\ \mathcal{V}_1(x,\hat{x}) = 0.9962 - 3.6481x + 1.3833\hat{x}-\\ 12.2918x^2 + 4.5125x\hat{x} - 0.3641\hat{x}^2 + 1.2333x^3-\\ 0.6245x^2\hat{x} + 0.1566x\hat{x}^2 + 0.0627\hat{x}^3, \text{ if } q = q_1;\\ \mathcal{V}_2(x,\hat{x}) = 0.8403 - 0.3703x - 0.3703\hat{x}-\\ 0.3874x^2 + 0.7815x\hat{x} - 0.3874\hat{x}^2 + 0.0693x^3-\\ 0.0518x^2\hat{x} - 0.0518x\hat{x}^2 + 0.0693\hat{x}^3, \text{ if } q = q_2;\\ \mathcal{V}_t(x,\hat{x}) = -1.3443 + 0.2775x + 0.2477\hat{x}-\\ 0.0944x^2 - 0.1661x\hat{x} - 0.0795\hat{x}^2 - 0.0249x^3+\\ 0.0256x^2\hat{x} + 0.0195x\hat{x}^2 - 0.0256\hat{x}^3, \text{ if } q = q_t;\end{cases}$$

Thus, system G is not approximate $(2, 1)$-prognosable according to Theorem 4.

VI. Conclusion

In this paper, we developed an abstraction-free approach to verify (the lack of) approximate (M, δ)-prognosability for discrete-time control systems based on barrier certificates. The verification of approximate (M, δ)-prognosability was reduced to a safety problem for the verification system, which can be checked via barrier certificates. Finally, we provided algorithms to search for valid barrier certificates based on the CEGIS framework. In the future, we would like to develop an algorithm to design (M, δ)-prognoser for approximate (M, δ)-prognosable systems.

References

[1] R. Ammour, E. Leclercq, E. Sanlaville, and D. Lefebvre. Fault prognosis of timed stochastic discrete event systems with bounded estimation error. *Automatica*, 82:35–41, 2017.

[2] R. Barcelos and J. Basilio. Disjunctive fault prediction of decentralized discrete event systems: Verification, predictor design and k-copredictability. *Automatica*, 148:110769, 2023.

[3] A. Chouchane and M. Ghazel. Fault-prognosability, k-step prognosis and k-step predictive diagnosis in partially observed petri nets by means of algebraic techniques. *Automatica*, 162:111513, 2024.

[4] L. De Moura and N. Bjørner. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.

[5] E. De Santis, M. Di Benedetto, Gabriella Fiore, and G. Pola. Approximate predictability of pseudo-metric systems. *Nonlinear Analysis: Hybrid Systems*, 36:100869, 2020.

[6] S. Diamond and S. Boyd. CVXPY: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

[7] S. Genc and S. Lafortune. Predictability of event occurrences in partially-observed discrete-event systems. *Automatica*, 45(2):301–311, 2009.

[8] P. Jagtap, S. Soudjani, and M. Zamani. Formal synthesis of stochastic systems via control barrier certificates. *IEEE Transactions on Automatic Control*, 66(7):3097–3110, 2020.

[9] Shadi Tasdighi Kalat, Siyuan Liu, and Majid Zamani. Modular verification of opacity for interconnected control systems via barrier certificates. *IEEE Control Systems Letters*, 6:890–895, 2021.

[10] D. Lefebvre. Fault diagnosis and prognosis with partially observed petri nets. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(10):1413–1424, 2014.

[11] S. Liu, A. Trivedi, X. Yin, and M. Zamani. Secure-by-construction synthesis of cyber-physical systems. *Annual Reviews in Control*, 53:30–50, 2022.

[12] S. Liu, X. Yin, and M. Zamani. On a notion of approximate opacity for discrete-time stochastic control systems. In *2020 American Control Conference (ACC)*, pages 5413–5418. IEEE, 2020.

[13] S. Liu and M. Zamani. Verification of approximate opacity via barrier certificates. *IEEE Control Systems Letters*, 5(4):1369–1374, 2020.

[14] Z. Ma, X. Yin, and Z. Li. Marking predictability and prediction in labeled petri nets. *IEEE Transactions on Automatic Control*, 66(8):3608–3623, 2021.

[15] V. Murali, S. Kalat, and M. Zamani. A data-driven approach to approximate opacity verification. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 5085–5090. IEEE, 2023.

[16] F. Nouioua, P. Dague, and L. Ye. Predictability in probabilistic discrete event systems. In *Soft Methods for Data Science*, pages 381–389. Springer, 2017.

[17] A. Papachristodoulou, J. Anderson, G. Valmorbida, S. Prajna, P. Seiler, P. Parrilo, M. Peet, and D. Jagt. SOSTOOLS version 4.00 sum of squares optimization toolbox for MATLAB. *arXiv preprint arXiv:1310.4716*, 2013.

[18] A. Peruffo, D. Ahmed, and A. Abate. Automated and formal synthesis of neural barrier certificates for dynamical models. In *International conference on tools and algorithms for the construction and analysis of systems*, pages 370–388. Springer, 2021.

[19] G. Pola, E. De Santis, and M. Di Benedetto. Approximate diagnosis of metric systems. *IEEE Control Systems Letters*, 2(1):115–120, 2017.

[20] G. Pola, E. De Santis, and M. Di Benedetto. Approximate current state observability of discrete-time nonlinear systems under cyber-attacks. *Nonlinear Analysis: Hybrid Systems*, 50:101403, 2023.

[21] S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *International Workshop on Hybrid Systems: Computation and Control*, pages 477–492. Springer, 2004.

[22] A. Solar-Lezama. *Program synthesis by sketching*. University of California, Berkeley, 2008.

[23] J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization methods and software*, 11(1-4):625–653, 1999.

[24] A. Watanabe, R. Sebem, A.B. Leal, and M. da S Hounsell. Fault prognosis of discrete event systems: An overview. *Annual Reviews in Control*, 51:100–110, 2021.

[25] X. Yin. Verification of prognosability for labeled petri nets. *IEEE Transactions on Automatic Control*, 63(6):1828–1834, 2018.

[26] X. Yin and Z. Li. Decentralized fault prognosis of discrete event systems with guaranteed performance bound. *Automatica*, 69:375–379, 2016.

[27] X. Yin, M. Zamani, and S. Liu. On approximate opacity of cyber-physical systems. *IEEE Transactions on Automatic Control*, 66(4):1630–1645, 2020.

[28] B. Zhong, W. Dong, X. Yin, and M. Zamani. Verification of diagnosability for cyber-physical systems via hybrid barrier certificates. In *8th IFAC Conference on Analysis and Design of Hybrid Systems (ADHS)*, 2024.