A Unified Framework for Verification of Observational Properties for Partially-Observed Discrete-Event Systems

Jianing Zhao, *Student Member, IEEE*, Shaoyuan Li, *Senior Member, IEEE*, and Xiang Yin, *Member, IEEE*

***Abstract*—In this article, we investigate property verification problems in partially-observed discrete-event systems (DES). Particularly, we are interested in verifying *observational properties* that are related to the information-flow of the system. Observational properties considered here include diagnosability, predictability, detectability, and opacity, which have drawn considerable attentions in the literature. However, in contrast to existing results, where different verification procedures are developed for different properties case by case, in this work, we provide a *unified framework* for verifying all these properties by reducing each of them as an instance of *HyperLTL model checking*. Our approach is based on the construction of a Kripke structure that effectively captures the issue of unobservability as well as the finite string semantics in partially-observed DES so that HyperLTL model checking techniques can be suitably applied. Then for each observational property considered, we explicitly provide the HyperLTL formula to be checked over the Kripke structure for the purpose of verification. Our approach is uniform in the sense that all different properties can be verified with the same model checking engine and also brings new insights for classifying observational properties in terms of their verification complexity. Numerical experiments are conducted, which show that our framework is computationally more efficient for verifying properties involving quantifier alternations, such as opacity, compared with the standard subset-based approaches.**

***Index Terms*—Discrete-event systems (DES), HyperLTL, partial observation, property verification.**

I. Introduction

Discrete-Event systems (DES) is an important class of complex engineering systems with discrete state spaces and event-triggering dynamics [6]. It is widely used in the modeling and analysis of the high-level logic behaviors of complex automated systems, such as manufacturing systems, softwares, and autonomous robots. Given a DES, one of the most fundamental problems is to determine whether or not the designed system satisfies some desired specifications of our interest by formal and algorithmic procedures. This is also referred to as the *property verification* problem, which is critical to ensure safety and security of DES [18].

Manuscript received 20 June 2023; revised 10 December 2023; accepted 6 January 2024. Date of publication 18 January 2024; date of current version 28 June 2024. This work was supported by the National Natural Science Foundation of China under Grant 62061136004, Grant 62173226, and Grant 61803259. Recommended by Associate Editor Cristian Mahulea. *(Corresponding author: Xiang Yin.)*

The authors are with the Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China, and also with the Key Laboratory of System Control and Information Processing, the Ministry of Education of China, Shanghai 200240, China (e-mail: jnzhao@sjtu.edu.cn; syli@sjtu.edu.cn; yinxiang@sjtu.edu.cn).

Digital Object Identifier 10.1109/TAC.2024.3355378

In many scenarios, DES are *partially-observed* either from the system-user's point of view due to the limited sensing capabilities, or from the outsider's point of view due to the partial information release [11]. In this context, one may need to determine whether or not the observer has sufficient knowledge about the system based on both the DES model and the partial observations. Such properties related to the *information-flow* of the partially-observed DES are referred to as the *observational properties*. In this article, we are concerned with the verifications of observational properties for partially observed DES.

Property verification of partially-observed DES dates back to the early investigations of supervisory control of partially-observed DES, where the notion of observability was investigated [17]. In this setting, it is usually assumed that the behaviors of the systems can only be observed partially via a natural projection or an observation mask, and one needs to determine whether or not the imperfect information is sufficient to realize a supervisor. Later on, verification of partially-observed DES has been investigated more thoroughly in the contexts of fault diagnosis, fault predication, state detection, and security analysis. See the recent textbook [11] and tutorial paper [29] for more details on this topic.

Here, we briefly review some important observational properties that are considered in this work as follows. The notion of diagnosability is one of the most widely investigated observational properties in DES literature [22], which characterizes the ability that one can always determine the occurrences of fault events within a finite delay. As the dual of the fault diagnosis problem, the property of prognosability or *predictability* was proposed in [10] and [13] for the fault prognosis problem, which provides the necessary and sufficient condition under which the fault can always be predicted with no false-alarm or miss-alarm. In the context of state estimation of partially-observed DES, Shu and Lin [23], [24], [25] proposed several different notions of *detectability* to characterize whether or not the system state can be determined unambiguously. More recently, motivated by the security and privacy considerations in cyber-physical systems, the notion of *opacity* has drawn many attentions, where it is assumed that there exists a passive intruder (eavesdropper) that can access the information-flow of the system and the system has some "secret" that does not want to be revealed to the intruder [5]. Depending on different security requirements, different notions of opacity have been studied including, e.g., initial-state opacity [21], current-state opacity [16], and infinite-step opacity [20].

While there is a wide literature on the verification of observational properties for partially-observed DES, several problems still remain. In particular, the existing approaches for the verification of partially-observed DES are mainly based on the observer structure and its variants [6]. For some properties, such as diagnosability, predictability, and strong detectability, researchers have further proposed polynomial-time algorithms [13], [14], [30]. However, the existing verification techniques are mainly developed for different properties *case by case*. The following questions arise naturally.

1) Can we provide a unified methodology for verifying the existing notions of observational properties in literature without investigating each of them case by case?

2) Can we find a suitable way to classify different notions of observational properties in literature in terms of their similarities and the verification complexity?

In this article, we aim to answer the above two questions by providing a unified approach for verifying partially-observed DES. Our approach relies on the recently developed new temporal logic called *HyperLTL* [7]. HyperLTL generalizes the standard linear-time temporal logic (LTL), which is evaluated over only a single trace, by adding quantifiers among different traces. HyperLTL has been shown as a very suitable tool for expressing information-flow properties (also called hyperproperties [8]) in the context of formal verification. Specifically, our uniform framework consists of two steps. First, for a DES plant model, we construct the corresponding (modified) *Kripke structure* that tracks both the state information and the observation information in the system. The issue of unobservability is effectively handled by the proposed structure. Next, we show that most of the observational properties in DES literature can be captured by explicit HyperLTL formulas over the constructed Kripke structure. These properties include, but not restricted to, diagnosability, predictability, (strong/weak/I-/delayed) detectability, and (initial-state/current-state/infinite-step) opacity.

Although verification algorithms already exist for these observational properties in literature, our unified approach is still of significance in threefold as follows.

1) First, our approach is uniform in the sense that all properties are expressed using the same logic over the same Kripke structure. As a consequence, one does not need to develop a customized verification algorithm for each property case by case any more.
2) Second, by expressing observational properties of DES in terms of HyperLTL, the proposed unified framework provides the access to HyperLTL model checking algorithms for the property verification, based on which one can leverage many highly optimized efficient tools, such as `MCHyper` [9], `HyperQube` [12], and `AutoHyper` [3]. Particularly, we show by numerical experiments that our unified approach is more efficient than the standard DES algorithms for the properties involving quantifier alternations, such as weak detectability and opacity.
3) Finally, by writing down each observational property explicitly in HyperLTL, our framework naturally provides a complexity hierarchy for different properties in terms of the alternation depth of the quantifiers.

We would like to remark that, although HyperLTL itself is a tool for specifying information-flow properties, it cannot be directly applied to check observational properties in DES due to the following two discrepancies. The first technical challenge is the presence of *unobservable events*. Specifically, observational properties are evaluated over the observation sequence to which an unobservable event does not contribute. This is different from the standard HyperLTL model checking where the time-indexes of the internal trace and its information-flow are the same. Second, the semantics of HyperLTL are defined over *infinite* traces while observational properties in DES are usually concerned with *finite* strings. For example, although initial-state opacity has been expressed using HyperLTL (not for DES models and without unobservable events) [1], [28], it has been pointed out by [18] that expressing current-state opacity or infinite-step opacity in terms of HyperLTL is technically challenging due to the fact that the quantification acts at the beginning of trajectories rather than every instant of trajectories. All these technical challenges in applying HyperLTL to DES have been addressed in our results.

Finally, we note that model checking techniques have already been used in literature for the verification of partially-observed DES. For example, model checking for diagnosability of DES is studied in [4], [19], and [27]. However, these works still use model checking over *single trace*, such as LTL model checking. In order to capture the *system-wide* requirements in observational properties, existing works need to build the information structure for the underlying specific property, such as twin-plant for diagnosability. However, here we use HyperLTL directly, which does not need to construct an information-synchronization structure for each specific property.

II. Preliminaries

A. System Model

Let Σ be a finite set of events (or alphabets). A finite (respectively, infinite) string $s=\sigma_1\cdots\sigma_n(\cdots)$, $\sigma_i\in\Sigma$ is a finite (respectively, infinite) sequence of events. We denote by Σ^* and Σ^ω the sets of all finite and infinite strings over Σ, respectively. The empty string ϵ is included in Σ^*. The length of a finite string $s\in\Sigma^*$ is denoted by $|s|$ and $|\epsilon|=0$. A $*$-language (respectively, ω-language) is a set of finite (respectively, infinite) strings. Given a language $L\subseteq\Sigma^*\cup\Sigma^\omega$, the *prefix closure* of language L, denoted by $\overline{L}$, is defined as the set of all its finite prefixes, i.e., $\overline{L}=\{s\in\Sigma^*:\exists w\in\Sigma^*\cup\Sigma^\omega \text{ s.t. } sw\in L\}$. A $*$-language $L\subseteq\Sigma^*$ is said to be prefix closed if $L=\overline{L}$. Given any string $s\in L$, the postlanguage of s in L is defined as $L/s=\{w\in\Sigma^*\cup\Sigma^\omega:sw\in L\}$.

We consider a DES modeled by a finite-state automaton (FSA)

$$G=(X,\Sigma,\delta,X_0),$$

where X is a finite set of states; Σ is a finite set of events; $\delta:X\times\Sigma\to X$ is a partial transition function such that: for any $x,x'\in X$ and $\sigma\in\Sigma$, $x'=\delta(x,\sigma)$ means that there exists a transition from state x to state x' with event label σ; and $X_0\subseteq X$ is the set of all possible initial states. We also extend the transition function to $\delta:X\times\Sigma^*\to X$ recursively by: 1) $\delta(x,\epsilon)=x$; and 2) for any $x\in X$, $s\in\Sigma^*$, and $\sigma\in\Sigma$, we have $\delta(x,s\sigma)=\delta(\delta(x,s),\sigma)$. We define $\mathcal{L}(G,x)=\{s\in\Sigma^*:\delta(x,s)!\}$ as the set of finite strings that can be generated by system G from state $x\in X$. For simplicity, we define $\mathcal{L}(G)=\cup_{x_0\in X_0}\mathcal{L}(G,x_0)$ as the $*$-language generated by system G. Similarly, we define $\mathcal{L}^\omega(G,x)=\{s\in\Sigma^\omega:\forall t\in\overline{\{s\}} \text{ s.t. } \delta(x,t)!\}$ as the infinite strings generated by system G starting from $x\in X$ and we also define $\mathcal{L}^\omega(G)=\cup_{x_0\in X_0}\mathcal{L}^\omega(G,x_0)$.

In a partially-observed DES, not all events can be observed perfectly. To this end, we consider an observation mask function

$$M:\Sigma\to\mathcal{O}\cup\{\epsilon\}$$

where $\mathcal{O}$ is the set of observation symbols. An event $\sigma\in\Sigma$ is said to be *unobservable* if $M(\sigma)=\epsilon$; otherwise, it is observable. We denote by Σ_o and Σ_{uo} the sets of observable and unobservable events, respectively. Moreover, events $\sigma,\sigma'\in\Sigma$ are said to be *indistinguishable* if $M(\sigma)=M(\sigma')$. The mask function is also extended to $M:\Sigma^*\cup\Sigma^\omega\to\mathcal{O}^*\cup\mathcal{O}^\omega$ such that, for any $s\in\mathcal{L}(G)$, we have 1) $M(\epsilon)=\epsilon$; and 2) for any $s\in\Sigma^*,\sigma\in\Sigma$, we have $M(s\sigma)=M(s)M(\sigma)$. We also extend mask to $M:2^{\Sigma^*\cup\Sigma^\omega}\to2^{\mathcal{O}^*\cup\mathcal{O}^\omega}$ by: for any $L\subseteq\Sigma^*\cup\Sigma^\omega:M(L)=\{M(s):s\in L\}$. Therefore, $M(\mathcal{L}(G))$ and $M(\mathcal{L}^\omega(G))$ are the observed $*$- and ω-languages generated by system G, respectively.

For simplicity, we make the following standard assumptions in the analysis of partially-observed DES.

A1 System G is live, i.e., $\forall x\in X,\exists\sigma\in\Sigma:\delta(x,\sigma)!$.

A2 System G does not contain an unobservable cycle, i.e., $\forall x\in X,\forall s\in\Sigma^*\setminus\{\epsilon\}:x=\delta(x,s)\Rightarrow M(s)\neq\epsilon$.

Since the system is partially-observed, the system-user needs to determine the state of the system based on the observation string, which is referred to as the *state estimation problem*. In this article, we will consider the following three types of state estimates. All properties of partially-observed DES in this article will be defined using state estimates [29].

Definition 1 (State Estimates): Let $\alpha\in M(\mathcal{L}(G))$ be an observation string. Then

1) the *initial-state estimate* upon observation α is the set of initial states the system could start from initially, i.e.,

$$\hat{X}_{G,0}(\alpha)=\{x_0\in X_0:\exists s\in\mathcal{L}(G,x_0) \text{ s.t. } M(s)=\alpha\} \quad (1)$$

2) the *current-state estimate* upon observation α is the set of states the system could be in currently, i.e.,

$$\hat{X}_G(\alpha)=\left\{\delta(x_0,s)\in X:\begin{array}{c}\exists x_0\in X_0, s\in\mathcal{L}(G,x_0)\\ \text{s.t. } M(s)=\alpha\end{array}\right\}. \quad (2)$$

Furthermore, let $\alpha\beta\in M(\mathcal{L}(G))$ be an observation string, where α is a prefix of the entire observation $\alpha\beta$. Then

1) the *delayed-state estimate* for the instant of α upon observation $\alpha\beta$ is the set of states the system could be in $|\beta|$ steps ago when $\alpha\beta$ is observed, i.e.,

$$\hat{X}_G(\alpha\mid\alpha\beta)=\left\{\delta(x_0,s)\in X:\begin{array}{c}\exists x_0\in X_0, sw\in\mathcal{L}(G,x_0)\text{ s.t.}\\ M(s)=\alpha\wedge M(sw)=\alpha\beta\end{array}\right\}. \quad (3)$$

B. LTL and HyperLTL

Let $\mathcal{AP}$ be a set of atomic propositions representing some basic properties of interest. A *trace* $\pi=\pi_0\pi_1\cdots\in(2^{\mathcal{AP}})^\omega$ is an infinite sequence over $2^{\mathcal{AP}}$. We denote by $\pi[i]=\pi_i$ the ith element in the trace and by $\pi[i,\infty]=\pi_i\pi_{i+1}\cdots\in(2^{\mathcal{AP}})^\omega$ its suffix starting from the ith instant. LTL is a widely used approach for evaluating whether or not a trace π satisfies some desired property. The syntax of LTL formulas is as follows:

$$\psi ::= a\mid\neg\psi\mid\psi\vee\psi\mid\bigcirc\psi\mid\psi\mathcal{U}\psi,$$

where $a\in\mathcal{AP}$ is an atomic proposition; $\neg$ and $\vee$ are Boolean operators "negation" and "disjunction," respectively, and $\bigcirc$ and $\mathcal{U}$ are temporal operators "next" and "until," respectively. We can induce Boolean operators, such as "implication" by $\psi_1\rightarrow\psi_2\equiv\neg\psi_1\vee\psi_2$ and "conjunction" by $\psi_1\wedge\psi_2\equiv\neg(\neg\psi_1\vee\neg\psi_2)$. Furthermore, we can induce temporal operators "eventually" by $\Diamond\psi\equiv\top\mathcal{U}\psi$ and "always" by $\Box\psi\equiv\neg\Diamond\neg\psi$. The semantics of LTL can be found in [2]. We denote by $\pi\models\varphi$ that trace π satisfies LTL formula φ.

Note that LTL can only evaluate a *single trace*. In many applications, the desired property is *system-wide* and can only be evaluated among multiple traces. For example, in diagnosability analysis, we need to check the existence of two strings with the same observation: one is fault but the other is normal. HyperLTL generalizes LTL by further supporting *trace quantifiers*. Let $\mathcal{V}=\{\pi_1,\pi_2,\ldots\}$ be the set of *trace variables*, where each π_i represents an individual trace. The syntax of HyperLTL formulas is as follows [7]:

$$\phi ::= \exists\pi.\,\phi\mid\forall\pi.\,\phi\mid\psi,$$
$$\psi ::= a^\pi\mid\neg\psi\mid\psi\vee\psi\mid\bigcirc\psi\mid\psi\mathcal{U}\psi$$

where $\exists$ and $\forall$ are the universal and the existential trace quantifiers, representing "for some trace" and "for all traces," respectively. Formula ψ is just an LTL formula except that the atomic propositions can refer to distinct trace variables. Particularly, since HyperLTL formulas can refer to multiple traces, we denote by a^π an atomic proposition $a\in\mathcal{AP}$ that should be checked on trace π.

The semantics of HyperLTL are defined over *a set of* traces $T\subseteq(2^{\mathcal{AP}})^\omega$ and a *partial* mapping (called trace assignment) $\Pi:\mathcal{V}\rightarrow(2^{\mathcal{AP}})^\omega$. In particular, we denote by $\Pi_\emptyset$ the empty assignment whose domain is the empty set $\emptyset$. We denote by $\Pi[\pi\mapsto\xi]$ the same trace assignment as Π expect that π is mapped to ξ. The trace assignment *suffix* $\Pi[i,\infty]$ denotes the trace assignment $\Pi'(\pi)=\Pi(\pi)[i,\infty]$ for all π. Then we denote by $(T,\Pi)\models\phi$ that HyperLTL ϕ is satisfied over a set of traces $T\subseteq(2^{\mathcal{AP}})^\omega$ and trace assignment $\Pi:\mathcal{V}\rightarrow(2^{\mathcal{AP}})^\omega$, which is defined as follows:

$$\begin{array}{lll}(T,\Pi)\models\exists\pi.\phi & \text{iff} & \exists\xi\in T:(T,\Pi[\pi\mapsto\xi])\models\phi\\ (T,\Pi)\models\forall\pi.\phi & \text{iff} & \forall\xi\in T:(T,\Pi[\pi\mapsto\xi])\models\phi\\ (T,\Pi)\models a^\pi & \text{iff} & a\in\Pi(\pi)[0]\\ (T,\Pi)\models\neg\psi & \text{iff} & (T,\Pi)\not\models\psi\\ (T,\Pi)\models\psi_1\vee\psi_2 & \text{iff} & (T,\Pi)\models\psi_1\text{ or }(T,\Pi)\models\psi_2\\ (T,\Pi)\models\bigcirc\psi & \text{iff} & (T,\Pi[1,\infty])\models\psi\\ (T,\Pi)\models\psi_1\mathcal{U}\psi_2 & \text{iff} & (\exists i\geq 0:(T,\Pi[i,\infty])\models\psi_2)\wedge\\ & & (\forall 0\leq j<i:(T,\Pi[j,\infty])\models\psi_1).\end{array}$$

We say a set of traces $T\subseteq(2^{\mathcal{AP}})^\omega$ satisfy a HyperLTL formula ϕ, denoted by $T\models\phi$, if $(T,\Pi_\emptyset)\models\phi$.

C. Kripke Structure

In model checking of HyperLTL, the set of traces $T\subseteq(2^{\mathcal{AP}})^\omega$ are usually generated by a *Kripke structure*. Formally, a Kripke structure is a tuple $K=(Q,Q_0,\Delta,\mathcal{AP},L)$, where Q is the set of states, $Q_0\subseteq Q$ is the set of initial states, $\Delta\subseteq Q\times Q$ is the transition relation, $\mathcal{AP}$ is the set of atomic propositions and $L:Q\rightarrow 2^{\mathcal{AP}}$ is the labeling function.

We say $\rho=\rho_0\rho_1\ldots\in Q^\omega$ is a *run* in K if $\rho_0\in Q_0$ and $\langle\rho_i,\rho_{i+1}\rangle\in\Delta,\forall i\geq 0$. We say $\pi=\pi_0\pi_1\ldots\in(2^{\mathcal{AP}})^\omega$ is a *trace* in K if there exists a run $\rho=\rho_0\rho_1\ldots\in Q^\omega$ such that $\pi_i=L(\rho_i),\forall i\geq 0$. We denote by $\mathsf{Run}(K)$ and $\mathsf{Trace}(K)$ the sets of all runs and traces generated by K, respectively. Then we say that a Kripke structure K satisfies HyperLTL formula ϕ, denoted by $K\models\phi$, if $\mathsf{Trace}(K)\models\phi$.

III. Partially-Observed DES in Kripke Structure

The main objective of this article is to use HyperLTL model checking techniques to solve the observational property verification problems for partially-observed DES. To this end, we need to transform the FSA model for DES into a Kripke structure for the purpose of model checking.

In HyperLTL model checking, atomic propositions are usually assigned to each state or transition in the system model. However, in the setting of partially-observed DES, we note that, for any internal string $s=\sigma_1\sigma_2\cdots\in\Sigma^\omega$, its observation string is $M(\alpha)=o_1o_2\cdots\in\mathcal{O}^\omega$, where for each $i\geq 1$, o_i is not necessarily the observation of event σ_i, since there may exist unobservable strings in between. Therefore, the time-indexes of the internal string and its information-flow may be mismatched. We also cannot assign the empty proposition to those unobservable transitions since it means "no property of interest," which is different from the empty observation.

To address unobservability, let $x,x'\in X$ be two states in G and $o\in\Delta\cup\{\epsilon\}$ be an observation symbol including the empty observation ϵ. We denote by $x\rightsquigarrow^o x'$ if x can reach x' via some string whose observation is o, i.e., $\exists s\in\mathcal{L}(G,x)$ such that $\delta(x,s)=x'$ and $M(s)=o$. Without unobservable event, the above string s must be a single event when $o\in\Delta$, and be ϵ when $o=\epsilon$. However, for the general case, s can be a string with more than one event even when $o=\epsilon$.

Now, we present how to construct the Kripke structure associated with a DES for the purpose of verification of observational properties.

Definition 2 (Kripke Structure for DES): Given partially-observed DES G with mask $M:\Sigma\rightarrow\mathcal{O}\cup\{\epsilon\}$, its associated Kripke structure is defined by

$$K_G=(Q,Q_0,\Delta,\mathcal{AP},L)$$

where

1) $Q\subseteq X\times(\mathcal{O}\cup\{\epsilon\})$ is the set of states;
2) $Q_0=\{(x,\epsilon)\in X\times\{\epsilon\}:\exists x_0\in X_0\text{ s.t. }x_0\rightsquigarrow^\epsilon x\}$ is the set of initial states;
3) $\Delta\subseteq Q\times Q$ is the transition function defined by: for any two states $q=(x,o),q'=(x',o')\in Q$, we have

$$\langle(x,o),(x',o')\rangle\in\Delta\text{ iff }x\overset{o'}{\rightsquigarrow}x'\wedge o'\in\mathcal{O}$$

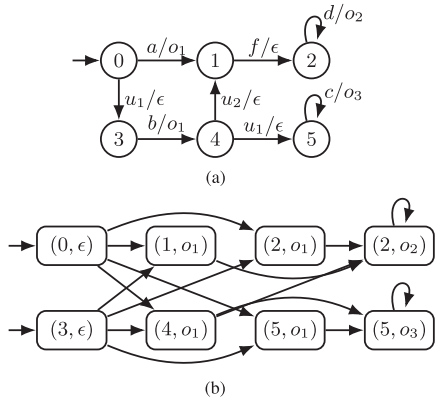

Fig. 1. Example for a partially-observed DES and its Kripke structure. (a) System G. (b) Kripke structure K_G.

4) $\mathcal{AP} = X \cup \mathcal{O}$ is the set of atomic propositions;
5) $L : Q \to 2^{\mathcal{AP}}$ is the labeling function defined by: for each $q = (x, o) \in Q$, we have

$$L(q) = \begin{cases} \{x\}, & \text{if } q \in Q_0 \\ \{x, o\}, & \text{if } q \notin Q_0. \end{cases} \tag{4}$$

Intuitively, for each state (x, o) in K_G, the first component x captures the current state of G and the second component o captures the latest observation symbol happened in G. At each state (x, o), when a feasible string $s \in \mathcal{L}(G, x)$ such that $M(s) = o'$ occurs, the Kripke structure moves to new state $(\delta(x, s), o')$, where the first component is determined by the transition function in G and the second component simply records the observed symbol. Note that the above string s may be in the form of $s = w_1\sigma w_2$, where $w_1, w_2 \in \Sigma_{uo}^*$ and $M(\sigma) = o$. Since there is no event observed initially, the initial state is of form (x, ϵ). The labeling function assigns the state symbol x and the observation symbol o as the atomic propositions hold at each (x, o). Therefore, the trace information in K_G already contains both the state sequence information and the observation sequence information, which are sufficient for the purpose of verifying observational properties of G. An example of the construction of K_G for G is given in Fig. 1 for the illustration.

To formally see the connection between G and K_G, let us consider an arbitrary infinite string $s \in \mathcal{L}^\omega(G, x_0)$ in G. Note that, we can always write s in the form of

$$s = w_0\sigma_1 w_1\sigma_2 w_2 \cdots \in \Sigma^\omega$$

where each $w_i \in \Sigma_{uo}^*$ is an unobservable string and each $\sigma_i \in \Sigma_o$ is an observable event with $M(\sigma_i) = o_i$. Let

$$\underbrace{x_0^0 \cdots x_0^{|w_0|}}_{\text{visited along } w_0} \underbrace{x_1^0 \cdots x_1^{|w_1|}}_{\text{visited along } \sigma_1 w_1} \underbrace{x_2^0 \cdots x_2^{|w_2|}}_{\text{visited along } \sigma_2 w_2} \cdots \in X^\omega \tag{5}$$

be the infinite sequence of states visited along s from x_0, where $x_0^0 = x_0$. Note that, in the construction of K_G, upon each observation, we will "jump" directly to a state without considering the states visited by unobservable strings in between. Therefore, we know that, *for any* of the indexes $k_0, k_1, \ldots$, where $k_i \in \{0, \ldots, |w_i|\}$, the following run exists in K_G:

$$\rho = (x_0^{k_0}, \epsilon)(x_1^{k_1}, o_1)(x_2^{k_2}, o_2) \cdots \in \mathsf{Run}(K_G).$$

We call such a run *compatible* with string s from initial state x_0. Since the choices of the indexes $k_0, k_1, \ldots$ are not unique, we denote by $\mathsf{Run}(s, x_0) \subseteq \mathsf{Run}(K_G)$ the set of all runs that are compatible with s and x_0. Note that when $s \in \mathcal{L}(G, x_0)$ is a finite string, there also exists a finite run that is compatible with s and x_0. With a slight abuse of notation, we still denote this by $\rho \in \mathsf{Run}(s, x_0)$.

On the other hand, for any run

$$\rho = (x_0, \epsilon)(x_1, o_1)(x_2, o_2) \cdots \in \mathsf{Run}(K_G)$$

by construction, we have $x_i \rightsquigarrow^{o_{i+1}} x_{i+1}$. Therefore, we can always find an initial state $\hat{x}_0 \in X_0$ and an infinite string $s \in \mathcal{L}^\omega(G, \hat{x}_0)$ such that $M(s) = o_1 o_2 \cdots$ and each x_i is reached by a prefix of s whose observation is $o_1 o_2 \cdots o_i$. That is, $\rho \in \mathsf{Run}(s, \hat{x}_0)$.

We illustrate the construction of Kripke structure K_G from DES G by the following example.

Example 1: Let us consider system G shown in Fig. 1(a), where $X = \{0, 1, 2, 3, 4, 5\}$, $\Sigma_o = \{a, b, c, d\}$, $\Sigma_{uo} = \{u_1, u_2, f\}$, $\mathcal{O} = \{o_1, o_2, o_3\}$ and the observation mask $M : \Sigma \to \mathcal{O}$ is defined by: $M(a) = M(b) = o_1$, $M(c) = o_3$, $M(d) = o_2$, and $M(u_1) = M(u_2) = M(f) = \epsilon$. The initial states of K_G are $(0, \epsilon)$ and $(3, \epsilon)$ since the system may reach state 3 from the initial state 0 via unobservable string u_1. From state $(0, \epsilon)$, by observing symbol o_1, one may reach states 1,2,4, and 5. Therefore, transitions from $(0, \epsilon)$ to states $(1, o_1), (2, o_1), (4, o_1)$, and $(5, o_1)$ are all defined in K_G. The labeling function can be encoded directly from the state, e.g., $L((0, \epsilon)) = \{0\}$ and $L((1, o_1)) = \{1, o_1\}$. For example, let us consider initial-state $x_0 = 0$ and infinite string $s = u_1 b u_2 f(d)^\omega \in \mathcal{L}^\omega(G)$ with $M(s) = o_1(o_2)^\omega$. Then a run in K_G compatible with s and x_0 can be, e.g., $(0, \epsilon)(4, o_1)(2, o_2)^\omega \in \mathsf{Run}(s, x_0)$ or $(3, \epsilon)(1, o_1)(2, o_2)^\omega \in \mathsf{Run}(s, x_0)$.

IV. Main Results

In this section, we consider the verification of all observational properties including diagnosability, predictability, detectability, and opacity in a unified manner using HyperLTL on the constructed Kripke structure for DES.

A. Diagnosability and Predictability in HyperLTL

For the fault diagnosis/prognosis problem, it is generally assumed that system G may contain some faults modeled as a set of fault events $\Sigma_F \subseteq \Sigma$. With a slight abuse of notation, for any string $s \in \Sigma \cup \Sigma^\omega$, we denote by $\Sigma_F \in s$ if string s contains a fault event in Σ_F. For simplicity, we assume that the state-space of G is partitioned as $X = X_N \dot\cup X_F$, where X_N is the set of normal states and X_F is the set of fault states such that

$$\forall x_0 \in X_0 \quad \forall s \in \mathcal{L}(G, x_0) : \Sigma_F \in s \Leftrightarrow \delta(x_0, s) \in X_F. \tag{6}$$

Note that this assumption is without loss of generality, since we can always refine the state-space of G such that the partition holds.

Diagnosability then characterizes whether or not we can always detect the occurrence of a fault event within a finite number of steps. To this end, we define

$$\Psi_F = \{s \in \mathcal{L}(G) : \Sigma_F \in s \wedge (\forall t \in \overline{\{s\}} \setminus \{s\} : \Sigma_F \notin t)\}$$

as the set of strings in which fault event occurs for the first time. The notion of diagnosability is reviewed as follows [22].

Definition 3 (Diagnosability): Given system G, observation mask $M : \Delta \to \mathcal{O} \cup \{\epsilon\}$ and fault events $\Sigma_F \subseteq \Sigma$, we say system G is *diagnosable* if any occurrence of fault can always be determined within a finite number of delays, i.e.,

$$(\exists n \in \mathbb{N})(\forall s \in \Psi_F)(\forall t \in \mathcal{L}(G)/s) \\ [|M(t)| \geq n \Rightarrow \hat{X}_G(M(st)) \subseteq X_F]. \tag{7}$$

When it refers to fault prediction, the objective is to predict the occurrences of fault events *in advance* such that

i) there is no miss-alarm in the sense that any fault can be alarmed before it occurs;
ii) there is no false-alarm in the sense that, once a fault alarm is issued, fault events will occur inevitably.

To define the notion of predictability, it is convenient to define

1) the set of *boundary states* $\partial(G)$, which is the set of normal states from which a fault event can occur in the next step, i.e.,

$$\partial(G) = \{x \in X_N : \exists \sigma \in \Sigma_F \text{ s.t. } \delta(x, \sigma)!\}$$

2) the set of *indicator states* $\mathfrak{I}(G)$, which is the set of normal states from which the system will enter fault states *inevitably* within a finite number of steps, i.e,

$$\mathfrak{I}(G) = \left\{ x \in X_N : \begin{array}{c} \exists n \in \mathbb{N} \quad \forall s \in \mathcal{L}(G,x) \\ \text{s.t. } |s| \geq n \Rightarrow \delta(x,s) \in X_F \end{array} \right\}.$$

Using the notions of boundary states and indicator states, we recall the definition of predictability as follows [10].

Definition 4 (Predictability): Given system G, observation mask $M : \Delta \to \mathcal{O} \cup \{\epsilon\}$, and fault events $\Sigma_F \subseteq \Sigma$, we say system G is *predictable* if the occurrence of fault can always be alarmed before it happens, i.e.,

$$\begin{aligned} &(\forall x_0 \in X_0)(\forall s \in \mathcal{L}(G,x_0) : \delta(x_0,s) \in \partial(G)) \\ &(\exists t \in \overline{\{s\}})[\hat{X}_G(M(t)) \subseteq \mathfrak{I}(G)]. \end{aligned} \tag{8}$$

Before expressing diagnosability in HyperLTL, we introduce some notation simplifications. For any trace variable π, we define F^π as the proposition that the trace is at a fault state, i.e., $F^\pi \equiv \bigvee_{x \in X_F} x^\pi$. Also, let π_1 and π_2 be two trace variables. Then we define

$$o^{\pi_1} = o^{\pi_2} \text{iff} \bigwedge_{o \in \mathcal{O}} o^{\pi_1} \leftrightarrow o^{\pi_2}$$

which is the proposition that π_1 and π_2 are observational-equivalent *at the initial instant*. As such, formula $\square(o^{\pi_1} = o^{\pi_2})$ represents that two infinite traces π_1, π_2 are observational-equivalent at any instants.

Now, we present the following theorem, stating how to formulate diagnosability of G using HyperLTL for Kripke structure K_G. Due to space constraints, all proofs are omitted in this article. All proofs as well as more illustrative examples can be found in [31].

Theorem 1 (HyperLTL for Diagnosability): System G is diagnosable if and only if $K_G \models \phi_{\text{dia}}$, where

$$\phi_{\text{dia}} = \forall \pi_1 . \forall \pi_2 . \left[\Diamond F^{\pi_1} \wedge \square(o^{\pi_1} = o^{\pi_2}) \rightarrow \Diamond F^{\pi_2} \right]. \tag{9}$$

Intuitively, the above theorem says that, to make the system diagnosable, for any two infinite strings having the same observation, if one string contains a fault event, i.e., $\Diamond F^{\pi_1}$, then the other string should also contain a fault event, i.e., $\Diamond F^{\pi_2}$. Otherwise, if $\square \neg F^{\pi_2}$, since $\square(o^{\pi_1} = o^{\pi_2})$, then the fault in the former string can never be determined within any finite number of steps.

We show the verification of diagnosability by an example.

Example 2: Let us still consider system G shown in Fig. 1(a) with Kripke structure K_G in Fig. 1(b). Here, we further assume $\Sigma_F = \{f\}$, i.e., $X_F = \{2\}$. One can observe that G is diagnosable since one can claim the occurrence of fault immediately after observing symbol o_2. Now, we show how this is captured by Theorem 1 using our framework. Taking the negation of ϕ_{dia}, we have

$$\neg\phi_{\text{dia}} = \exists \pi_1 . \exists \pi_2 . \left[\Diamond F^{\pi_1} \wedge \square(o^{\pi_1} = o^{\pi_2}) \wedge \square \neg F^{\pi_2} \right].$$

To satisfy F^{π_1}, trace π_1 must be of form $\pi_1 = \cdots \{2, o_2\}^\omega$, while to satisfy $\square \neg F^{\pi_2} = \neg \Diamond F^{\pi_2}$, trace π_2 must be of form $\pi_2 = \cdots \{5, o_3\}^\omega$. However, this implies that $\square(o^{\pi_1} = o^{\pi_2})$ cannot be further satisfied. Therefore, we have $K_G \models \phi_{\text{dia}}$.

For the case of predictability, we observe that, a system is *not* predictable if for some string that goes to a boundary state $x_1 \in \partial(G)$, there exists another string that goes to a normal but nonindicator state $x_2 \in X_N \setminus \mathfrak{I}(G)$ such that they have the same observation. This is because, from the former state x_1, a fault event can occur immediately, while from the latter state x_2, some nonfault string can still execute infinitely. In the context of traces in Kripke structure K_G, the former string can be captured by a trace π_1 such that $\Diamond F^{\pi_1}$, while the second string can be captured by a trace π_2 such that $\neg \Diamond F^{\pi_2}$. Furthermore, the observation equivalence condition is only applied *before* the first occurrence of fault in π_1. Therefore, $\Diamond F^{\pi_1}$ and the truncated observation equivalence can be captured together by $(o^{\pi_1} = o^{\pi_2}) \mathcal{U} F^{\pi_1}$. This suggests that G is not predictable if

$$\exists \pi_1 . \exists \pi_2 . \left[(o^{\pi_1} = o^{\pi_2}) \mathcal{U} F^{\pi_1} \wedge \neg \Diamond F^{\pi_2} \right].$$

Then by taking the negation of the existence of such two traces, we obtain the following main theorem for predictability.

Theorem 2 (HyperLTL for Predictability): System G is predictable if and only if $K_G \models \phi_{\text{pre}}$, where

$$\phi_{\text{pre}} = \forall \pi_1 . \forall \pi_2 . \left[(o^{\pi_1} = o^{\pi_2}) \mathcal{U} F^{\pi_1} \rightarrow \Diamond F^{\pi_2} \right]. \tag{10}$$

B. Detectability in HyperLTL

Detectability is a property characterizing whether or not the precise state of the system can be determined unambiguously under imperfect observations. Depending on the specific detection requirements, various notions of detectability have been proposed in literature. In this section, we consider variants of detectability including I-detectability, strong detectability, weak detectability, and delayed detectability, and show how each of them can be formulated in terms of the HyperLTL formula.

First, we review some existing notions of detectability.

Definition 5 (Detectability): Given system G and observation mask $M : \Sigma \to \mathcal{O} \cup \{\epsilon\}$, we say system G is

1) *I-detectable* [23] if the initial-state of the system can always be determined after a finite number of observations, i.e.,

$$(\exists n \in \mathbb{N})(\forall \alpha \in M(\mathcal{L}(G)) : |\alpha| \geq n)[|\hat{X}_{G,0}(\alpha)| = 1].$$

2) *Strongly detectable* [25] if the current-state of the system can always be determined after a finite number of observations, i.e.,

$$(\exists n \in \mathbb{N})(\forall \alpha \in M(\mathcal{L}(G)) : |\alpha| \geq n)[|\hat{X}_G(\alpha)| = 1].$$

3) *Weakly detectable* [25] if the current-state and the subsequent states of the system can be determined after a finite number of observations for some trajectory of the system, i.e.,

$$\begin{aligned} &(\exists n \in \mathbb{N})(\exists \alpha \in M(\mathcal{L}^\omega(G))) \\ &\qquad (\forall \beta \in \overline{\{\alpha\}} : |\beta| \geq n)[|\hat{X}_G(\beta)| = 1]. \end{aligned}$$

4) *Delayed-detectable* [24] if the precise state of the system at any instant can be determined after some observation delays, i.e.,

$$(\exists n \in \mathbb{N})(\forall \alpha\beta \in M(\mathcal{L}(G)) : |\beta| \geq n)[|\hat{X}_G(\alpha \mid \alpha\beta)| = 1].$$

Now, we formulate the above variants of detectability using HyperLTL. Based on different types of state-estimates, we present our result in three parts in what follows.

Still, before expressing detectability in HyperLTL, we define some notation simplifications for HyperLTL formulas. For any trace variable π, we define X_0^π as the proposition that the trace starts from an initial state in G, i.e., $X_0^\pi \equiv \bigvee_{x \in X_0} x^\pi$. Furthermore, for trace variables π_1 and π_2, we define $x^{\pi_1} = x^{\pi_2}$ as the proposition that the states of π_1 and π_2 at the initial instant are equivalent, i.e.,

$$x^{\pi_1} = x^{\pi_2} \text{ iff } \bigwedge_{x \in X} x^{\pi_1} \leftrightarrow x^{\pi_2}$$

We denote by $x^{\pi_1} \neq x^{\pi_2}$ if $\neg(x^{\pi_1} = x^{\pi_2})$, which means that $\pi_1[0]$ and $\pi_2[0]$ are not state-equivalent.

The following theorem states how to formulate I-detectability of G using HyperLTL for Kripke structure K_G.

Theorem 3 (HyperLTL for I-Detectability): System G is I-detectable if and only if $K_G \models \phi_{id}$, where

$$\phi_{id} = \forall\pi_1.\forall\pi_2.\begin{bmatrix}[X_0^{\pi_1} \wedge X_0^{\pi_2} \wedge \square(o^{\pi_1} = o^{\pi_2})] \\ \rightarrow (x^{\pi_1} = x^{\pi_2})\end{bmatrix}. \quad (11)$$

Intuitively, the above theorem says that, for any two infinite traces in K_G that are initiated from actual initial-states in G, if they always have the same observation proposition, then they must have the same state proposition initially. Otherwise, there exist two infinite traces starting from two distinct initial-states but having the same observation, which violates the requirement of I-detectability.

Next, we consider the cases of strong detectability and weak detectability. Compared with I-detectability, where two infinite traces need to have the same state proposition initially, strong detectability requires that two infinite traces need to *converge to* the same state proposition. Such a convergence requirement can be captured by the combination of temporal operators "*always eventually*" $\lozenge\square$. Recall that, in LTL, $\pi \models \lozenge\square\varphi$, if there exists $i \geq 0$ such that for any $j \geq i$, we have $\pi[j,\infty] \models \varphi$.

Now we present the following theorem stating how to formulate strong detectability using HyperLTL.

Theorem 4 (HyperLTL for Strong Detectability): System G is strongly detectable if and only if $K_G \models \phi_{sd}$, where

$$\phi_{sd} = \forall\pi_1.\forall\pi_2.\,[\square(o^{\pi_1} = o^{\pi_2}) \rightarrow \lozenge\square(x^{\pi_1} = x^{\pi_2})]\,. \quad (12)$$

The case of weak detectability is similar to the strong counterpart. The main difference is that, for strong detectability, we require that *for all* traces, we can eventually determine its state, while weak detectability only requires the *existence* of such a trace. Therefore, the HyperLTL condition ϕ_{wd} for weak detectability simply replaces the first universal quantifier $\forall$ in ϕ_{sd} by an existential quantifier $\exists$. Note that, although Theorem 5 seems to be similar to Theorem 4, there is significant difference here: $\forall.\forall.$ in ϕ_{sd} does not require quantifier alternation, while $\exists.\forall.$ in ϕ_{wd} has one time of quantifier alternation.

Theorem 5 (HyperLTL for Weak Detectability): System G is weakly detectable if and only if $K_G \models \phi_{wd}$, where

$$\phi_{wd} = \exists\pi_1.\forall\pi_2.\,[\square(o^{\pi_1} = o^{\pi_2}) \rightarrow \lozenge\square(x^{\pi_1} = x^{\pi_2})]\,. \quad (13)$$

Finally, we consider the case of delayed detectability, which seems to be more complicated since delayed-state estimate $\hat{X}_G(\alpha \mid \alpha\beta)$ is involved. However, we show that it can still be captured by HyperLTL quite elegantly in a similar fashion as the cases of other notions of detectability. To this end, we observe that, system G is not delayed-detectable if there exists an observation α such that $|\hat{X}_G(\alpha \mid \alpha\beta)| \geq 2$ no matter how long the future observation β is. Then by extending observation β to the infinite instant, we can obtain two infinite strings in G such that 1) they have the same observation; and 2) they reach different states at instant $|\alpha|$. This key observation leads to the following theorem.

Theorem 6 (HyperLTL for Delayed Detectability): System G is delayed-detectable if and only if $K_G \models \phi_{dd}$, where

$$\phi_{dd} = \forall\pi_1.\forall\pi_2.\,[\square(o^{\pi_1} = o^{\pi_2}) \rightarrow \square(x^{\pi_1} = x^{\pi_2})]\,. \quad (14)$$

C. Opacity in HyperLTL

Opacity is another important information-flow property describing the privacy and security requirements of the system. In this context, it is assumed that there exists an intruder (passive observer) that can also observe the occurrences of events through the observation mask. Furthermore, it is assumed that the system has some "secret". Then opacity captures the confidentiality that the secret can be revealed to the intruder via the information-flow. In the context of DES, the secret of the system is usually modeled as a set of secret states $X_S \subseteq X$. This naturally partitions the state space as $X = X_S \dot{\cup} X_{NS}$, where $X_{NS} = X \backslash X_S$ is the set of nonsecret states. According to what kind of secrets the system wants to protect, variants of opacity have been proposed in literature. In this section, we consider the initial-state opacity, infinite-step opacity, and current-state opacity, and formulate all of them in HyperLTL.

Definition 6 (Opacity): Given system G, observation mask $M : \Delta \rightarrow \mathcal{O} \cup \{\epsilon\}$, and secret states $X_S \subseteq X$, we say system G is

1) *initial-state opaque* [21], if the intruder can never know for sure that the system was initially from a secret state, i.e.,
$$(\forall\alpha \in M(\mathcal{L}(G)))[\hat{X}_{G,0}(\alpha) \not\subseteq X_S]$$
2) *current-state opaque* [16], if the intruder can never know for sure that the system is currently at a secret state, i.e.,
$$(\forall\alpha \in M(\mathcal{L}(G)))[\hat{X}_{G}(\alpha) \not\subseteq X_S]$$
3) *infinite-step opaque* [20], if the intruder can never know for sure that the system was at a secret state for any specific instant, i.e.,
$$(\forall\alpha\beta \in M(\mathcal{L}(G)))[\hat{X}_{G,0}(\alpha \mid \alpha\beta) \not\subseteq X_S].$$

For simplicity, we denote by S^π and NS^π the propositions that the trace is at a secret state and a nonsecret state, respectively, i.e., $S^\pi \equiv \bigvee_{x\in X_S} x^\pi$ and $NS^\pi \equiv \bigvee_{x\in X_{NS}} x^\pi$. Now, we show how these three variants of opacity can be formulated in terms of HyperLTL.

Essentially, initial-state opacity requires that, for any string, if it is initiated from a secret state, then there must exist another string such that 1) it is initiated from a nonsecret state; and 2) the two strings have the same observation. This requirement can be captured easily by HyperLTL formula based on the Kripke structure K_G as follows.

Theorem 7 (HyperLTL for Initial-State Opacity): System G is initial-state opaque if and only if $K_G \models \phi_{\text{iso}}$, where

$$\phi_{\text{iso}} = \forall\pi_1.\exists\pi_2.\begin{bmatrix}[X_0^{\pi_1} \wedge S^{\pi_1}] \rightarrow \\ [X_0^{\pi_2} \wedge \square(o^{\pi_1} = o^{\pi_2}) \wedge NS^{\pi_2}]\end{bmatrix}. \quad (15)$$

When it refers to current-state opacity, however, the following difficulty arises if we want to write down HyperLTL formula that is checked on Kripke structure K_G. Specifically, current-state opacity requires that for any *finite string* that ends up with a secret state, there exists another finite string ending up with a nonsecret state such that they have the same observation. However, the semantics of HyperLTL are defined over *infinite traces*. To capture this using HyperLTL, we need some mechanism to *indicate* that two infinite traces are at secret and nonsecret states, respectively, at the *same instant*. Furthermore, the observation equivalence requirement should be only applied up to that indicator instant, not for the entire infinite horizon.

In order to bridge the gap between the finite requirement in current-state opacity and the infinite semantics of HyperLTL, we modifies K_G by allowing the process to stop at any finite instant.

Definition 7 (Modified Kripke Structure): Given Kripke structure $K_G = (Q, Q_0, \Delta, \mathcal{AP}, L)$, we defined the *modified Kripke structure*

$$\tilde{K}_G = (\tilde{Q}, \tilde{Q}_0, \tilde{\Delta}, \widetilde{\mathcal{AP}}, \tilde{L})$$

where

1) $\tilde{Q} = Q \cup Q^c$ is the set of states, where $Q^c = \{(x^c, o^c) : (x, o) \in Q\}$ is simply a copy of the original state set Q;
2) $\tilde{Q}_0 = Q_0$ is the set of initial states;
3) $\tilde{\Delta} \subseteq \tilde{Q} \times \tilde{Q}$ is the transition function defined as follows:
   a) for any $q, q' \in Q : \langle q, q' \rangle \in \tilde{\Delta}$, we have $\langle q, q' \rangle \in \tilde{\Delta}$;
   b) for any $q = (x, o) \in Q$, we have
$$\langle (x, o), (x^c, o^c) \rangle, \langle (x^c, o^c), (x, o) \rangle \in \tilde{\Delta}. \tag{16}$$
4) $\widetilde{\mathcal{AP}} = X \cup \mathcal{O} \cup \{\tau\}$ is the set of atomic propositions, where τ is a new symbol;
5) $\tilde{L} : \tilde{Q} \to 2^{\widetilde{\mathcal{AP}}}$ is the labeling function defined by
   a) for any $q \in Q$, we have $\tilde{L}(q) = L(q)$;
   b) for any $q^c = (x^c, o^c) \in Q^c$, we have $\tilde{L}(q^c) = \{x, \tau\}$.

Specifically, we simply add a new copy state (x^c, o^c) for each state (x, o) in K_G. In addition to the original transitions in K_G, each state (x, o) and its copy state (x^c, o^c) form a loop. Furthermore, for each copy state, we assign it a new atomic proposition τ. Intuitively, τ will be used as an *indicator* to locate the specific instant of our interest for checking secret status.

To formally see this, consider an arbitrary run in K_G

$$\rho = (x_0, \epsilon)(x_1, o_1) \cdots \in \mathsf{Run}(K_G).$$

Then for any instant $k \geq 0$, based on the construction of $\tilde{K}_G$, there exists the following run in $\tilde{K}_G$:

$$\tilde{\rho} = (x_0, \epsilon)(x_1, o_1) \cdots \underbrace{(x_k, o_k)(x_k^c, o_k^c)(x_k, o_k)}_{\text{loop to copy state at instant k}} (x_{k+1}, o_{k+1}) \cdots$$

whose trace is given by

$$\tilde{L}(\tilde{\rho}) = \{x_0\}\{x_1, o_1\} \cdots \{x_k, o_k\}\{x_k, \tau\}\{x_k, o_k\} \cdots$$

As such, with the help of atomic proposition τ, we can easily locate x_k for which we what to check whether or not $x_k \in X_S$, and truncate the infinite sequence after τ.

Note that, in the above infinite trace, we only want to loop at the copy state *once* at the specific instant of interest. However, temporal operator $\Diamond\tau$ is not sufficient to express this since τ may occur multiple times. To this end, we define operator $\Diamond_1$ as "eventually and only once" as follows:

$$\Diamond_1 \tau \equiv \Diamond\tau \wedge \Box(\tau \to \bigcirc\Box\neg\tau) \tag{17}$$

i.e., τ will eventually occur and once it occurs, it will never occur in the future. Illustrative example for the construction of the modified Kripke structure can also be found in [31].

We formulate current-state opacity as a HyperLTL formula on $\tilde{K}_G$.

Theorem 8 (HyperLTL for Current-State Opacity): System G is current-state opaque if and only if $\tilde{K}_G \models \phi_{\text{cso}}$, where

$$\phi_{\text{cso}} = \forall\pi_1.\exists\pi_2. \left[\begin{array}{l}[\Diamond_1\tau^{\pi_1} \wedge \Box(\tau^{\pi_1} \to S^{\pi_1})] \to \\ [(o^{\pi_1} = o^{\pi_2})\mathcal{U}\tau^{\pi_1} \wedge \Box(\tau^{\pi_1} \to (\tau^{\pi_2} \wedge NS^{\pi_2}))]\end{array}\right]. \tag{18}$$

Formula ϕ_{cso} above is explained as follows. For trace π_1, which is quantified by the universal quantifier, we require that it visits a copy state only once, i.e., $\Diamond_1\tau^{\pi_1}$, and the copy state it visits is a secret state, i.e., $\Box(\tau^{\pi_1} \to S^{\pi_1})$. Then, for such an arbitrary π_1, we require the existence of trace π_2 such that 1) it has the same observation with π_1 until the stopping instant, i.e., $(o^{\pi_1} = o^{\pi_2})\mathcal{U}\tau^{\pi_1}$; and 2) when it stops, it is at a nonsecret copy state, i.e., $\Box(\tau^{\pi_1} \to (\tau^{\pi_2} \wedge NS^{\pi_2}))$.

The case of infinite-step opacity is similar to the case of current-state opacity. Specifically, it also requires that, for any string ending up with a secret state, there exists another string ending up with a nonsecret state such that they have the same observation. However, in addition, we need to further ensure that, for any string starting from the above secret, there exists another string starting from the above nonsecret state such that they have the same observation. Otherwise, the intruder may realize that the system was at a secret state after some steps. To capture this difference, one can simply replace the "truncated" observation equivalence condition $(o^{\pi_1} = o^{\pi_2})\mathcal{U}\tau^{\pi_1}$ in (18) by an infinite horizon version of observation equivalence condition $\Box(o^{\pi_1} = o^{\pi_2})$. This leads to the following theorem.

Theorem 9 (HyperLTL for Infinite-Step Opacity): System G is infinite-step opaque if and only if $\tilde{K}_G \models \phi_{\text{ifo}}$, where

$$\phi_{\text{ifo}} = \forall\pi_1.\exists\pi_2. \left[\begin{array}{l}[\Diamond_1\tau^{\pi_1} \wedge \Box(\tau^{\pi_1} \to S^{\pi_1})] \to \\ [\Box(o^{\pi_1} = o^{\pi_2}) \wedge \Box(\tau^{\pi_1} \to (\tau^{\pi_2} \wedge NS^{\pi_2}))]\end{array}\right]. \tag{19}$$

V. Discussions and Numerical Experiments

Now, let us go back to the second question in the introduction that why some properties are similar while some are more different. This, in fact, can be easily explained by the theory of HyperLTL. In HyperLTL, the *alternation depth* is referred to as the number of times the quantifiers alter from existential to universal, or vice versa. It is known that the verification complexity of HyperLTL will increase an exponential level when the formula has one more quantifier alternation [7]. For example, the alternation depth of "$\forall.\exists.$" is one, while the alternation depth of "$\forall.\forall.$" is zero. The former corresponds to the case of opacity and weak detectability, while the latter corresponds to other notions, such as diagnosability. Therefore, expressing observational properties in terms of HyperLTL also suggests a natural way to classify existing notions of observational properties in partially-observed DES: the larger alternation depth the property has, the higher verification complexity it will require.

Here, we present numerical experiments to compare the efficiency of the proposed HyperLTL-based framework with the customized DES algorithms for verifying diagnosability (DIAG), predictability (PRED), I-detectability (ID), strong/weak detectability (SD/WD), delayed detectability (DD), initial/current-state opacity (ISO/CSO), and infinite-step opacity (IFO). Specifically, we implement the DES algorithms as well as the as transformation algorithm from automata to Kripke structures in `Python3`. For DIAG, PRED, ID, SD, and DD, we adopt the twin-plant-based polynomial-time algorithms [29], and for WD, ISO, CSO, and IFO, we adopt the observer-based algorithms since they are inherently PSPACE-hard [29]. For the HyperLTL verification, we employ the newly developed model checker `AutoHyper` [3]. The experiments are conducted on a MacBook (8 G/256 G, Apple M1). Specifically, for each property with a fixed number of states, we randomly generate 50 systems and verify the property by both the customized DES algorithm and the HyperLTL-based algorithm; and we increase the state number from 15 to 80 states. Table I shows the average running times (in seconds) for each property with different sizes. All codes are available in the project website https://github.com/jnzhaooo/jtudes.

In the conducted experiments, we note that the first five properties involve no quantifier alternation, and they all have polynomial DES algorithms. For these properties, the HyperLTL-based approach does not seem to have computational advantage. For the last four properties, they all have one degree of quantifier alternation, and their DES algorithms are generally based on the subset construction, which requires exponential complexity. For these properties, however, the average running time of HyperLTL-based framework is considerably smaller. The main reason for the performance discrepancy is that HyperLTL model

TABLE I
STATISTIC RESULTS FOR NUMERICAL EXPERIMENTS ON RANDOMLY GENERATED SYSTEMS

	DIAG		PRED		ID		SD		DD		WD		ISO		CSO		IFO	
$\|X\|$	DES	HYP	DES	HYP	DES	HYP	DES	HYP	DES	HYP	DES	HYP	DES	HYP	DES	HYP	DES	HYP
15	0.03	0.43	0.06	0.41	0.13	0.23	0.08	1.53	0.35	0.75	3.41	2.93	3.74	0.72	2.23	0.85	4.68	0.97
30	0.08	0.85	0.07	0.68	0.17	0.54	0.09	4.72	1.26	2.41	10.4	3.48	9.44	1.12	6.14	1.60	24.6	2.17
50	0.14	1.07	0.27	1.15	0.51	2.06	0.23	7.35	3.38	4.02	20.9	5.42	18.4	1.45	12.9	3.16	73.9	4.18
80	0.61	1.48	0.46	1.32	1.39	3.13	0.87	12.9	5.62	6.93	53.2	6.36	26.1	1.94	32.3	5.73	171.3	7.21

checkers, such as `AutoHyper`, handle quantifier alternations more efficiently. Specifically, for the last four properties, where quantifier alternations are involved, the HyperLTL model checker reduces the verification problem to a language inclusion problem over an unfolded automaton and the specification automaton without explicitly searching the entire state space, which yield less complexity than the exponential-time worst-case. In summary, the experimental results show that the proposed framework not only provides a unified method but also is more efficient for properties that previously need to be checked using subset construction.

VI. CONCLUSION

In this article, we revisited the problems of verifying observational properties for partially-observed DES, which have been studied very actively and extensively in the past two decades in the context of DES. We showed that the recent developed new temporal logic called HyperLTL can be used as a suitable tool for unifying many of the important observational properties in literature. Our framework does not provide new decidability results since for all properties considered here, verification algorithms have already been developed. However, we believe that our unified view in terms of HyperLTL provides new insights for those properties that were previously investigated separately. Furthermore, our unified framework is of practical value since it provides the access to many of the efficient model checking tools for the purpose of verifying all these observational properties instead of developing a customized algorithm for each case.

We would like to remark that, although we have shown that many of the important properties in partially-observed DES can be formulated in terms of HyperLTL, there still exist properties that cannot be captured by the existing proposed framework. Examples of such properties are A-diagnosability [26] and A-detectability [15]. Similarly, for nonobservational properties, nonblockingness also cannot be expressed by HyperLTL. Essentially, these properties requiring the existence of a path from a state satisfying some condition is essentially a *branching-time property* which is beyond the semantics of HyperLTL. To address this issue and to further generalize our framework, a possible future direction is to use more expressive temporal logic, such as HyperCTL* [7] that supports both linear-time and branching-time properties over multiple traces.

REFERENCES

[1] M. Anand, V. Murali, A. Trivedi, and M. Zamani, "Formal verification of hyperproperties for control systems," in *Proc. Workshop Comput.-Aware Algorithmic Des. Cyber-Physical Syst.*, 2021, pp. 29–30.

[2] C. Baier and J.-P. Katoen, *Principles of Model Checking*. Cambridge, MA, USA: MIT Press, 2008.

[3] R. Beutner and B. Finkbeiner, "AutoHyper: Explicit-state model checking for HyperLTL," in *Proc. Int. Conf. Tools Algorithms Construct Anal. Syst.*, 2023, pp. 145–163.

[4] A. Boussif and M. Ghazel, "Diagnosability analysis of input/output discrete-event systems using model-checking," *IFAC-PapersOnLine*, vol. 48, no. 7, pp. 71–78, 2015.

[5] J. W. Bryans, M. Koutny, L. Mazaré, and Y. A. Ryan, "Opacity generalised to transition systems," *Int. J. Inf. Secur.*, vol. 7, no. 6, pp. 421–435, 2008.

[6] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. vol. 3. Berlin, Germany: Springer, 2021.

[7] M. R. Clarkson, B. Finkbeiner, M. Koleini, K. K. Micinski, M. N. Rabe, and C. Sánchez, "Temporal logics for hyperproperties," in *Proc. Int. Conf. Princ. Secur. Trust*, 2014, pp. 265–284.

[8] M. R. Clarkson and F. B. Schneider, "Hyperproperties," *J. Comput. Secur*, vol. 18, no. 6, pp. 1157–1210, 2010.

[9] B. Finkbeiner, M. N. Rabe, and C. Sánchez, "Algorithms for model checking HyperLTL and HyperCTL*," in *Proc. Int. Conf. Comput. Aided Verification*, 2015, pp. 30–48.

[10] S. Genc and S. Lafortune, "Predictability of event occurrences in partially-observed discrete-event systems," *Automatica*, vol. 45, no. 2, pp. 301–311, 2009.

[11] C. N. Hadjicostis. *Estimation and Inference in Discrete Event Systems*. Berlin, Germany: Springer, 2020.

[12] T.-H. Hsu, C. Sánchez, and B. Bonakdarpour, "Bounded model checking for hyperproperties," in *Proc. Int. Conf. Tools Algorithms Construction Anal. Syst.*, 2021, pp. 94–112.

[13] T. Jéron, H. Marchand, S. Genc, and S. Lafortune, "Predictability of sequence patterns in discrete event systems," *IFAC Proc.*, vol. 41, no. 2, pp. 537–543, 2008.

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

[15] C. Keroglou and C. N. Hadjicostis, "Detectability in stochastic discrete event systems," *Syst. Control Lett.*, vol. 84, pp. 21–26, 2015.

[16] F. Lin, "Opacity of discrete event systems and its applications," *Automatica*, vol. 47, no. 3, pp. 496–503, 2011.

[17] F. Lin and W. M. Wonham, "On observability of discrete-event systems," *Inf. Sci.*, vol. 44, no. 3, pp. 173–198, 1988.

[18] S. Liu, A. Trivedi, X. Yin, and M. Zamani, "Secure-by-construction synthesis of cyber-physical systems," *Annu. Rev. Control*, vol. 53, pp. 30–50, 2022.

[19] Y. Pencolé and A. Subias, "Diagnosability of event patterns in safe labeled time petri nets: A model-checking approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 2, pp. 1151–1162, Apr. 2022.

[20] A. Saboori and C. N. Hadjicostis, "Verification of infinite-step opacity and complexity considerations," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1265–1269, May 2012.

[21] A. Saboori and C. N. Hadjicostis, "Verification of initial-state opacity in security applications of discrete event systems," *Inf. Sci.*, vol. 246, pp. 115–132, 2013.

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

[23] S. Shu and F. Lin, "I-detectability of discrete-event systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 1, pp. 187–196, Jan. 2013.

[24] S. Shu and F. Lin, "Delayed detectability of discrete event systems," *IEEE Trans. Autom. Control*, vol. 58, no. 4, pp. 862–875, Apr. 2013.

[25] S. Shu, F. Lin, and H. Ying, "Detectability of discrete event systems," *IEEE Trans. Autom. Control*, vol. 52, no. 12, pp. 2356–2359, Dec. 2007.

[26] D. Thorsley and D. Teneketzis, "Diagnosability of stochastic discrete-event systems," *IEEE Trans. Autom. Control*, vol. 50, no. 4, pp. 476–492, Apr. 2005.

[27] T. M. Tuxi, L. K. Carvalho, E. V. L. Nunes, and A. E. C. da Cunha, "Diagnosability verification using LTL model checking," *Discrete Event Dyn. Syst.*, vol. 32, no. 3, pp. 399–433, 2022.

[28] Y. Wang, S. Nalluri, and M. Pajic, "Hyperproperties for robotics: Planning via HyperLTL," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 8462–8468.

[29] X. Yin, "Estimation and verification of partially-observed discrete-event systems," in *Wiley Encyclopedia of Electrical and Electronics Engineering*, Hoboken, NJ, USA: Wiley, 2019.

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

[31] J. Zhao, S. Li, and X. Yin, "Complete version of 'A unified framework for verification of observational properties for partially-observed discrete-event systems'," 2022, *arXiv:2205.01392*.