Protect Your Knowledge: Epistemic Property Enforcement of Discrete-Event Systems With Asymmetric Information

Shaowen Miao, *Graduate Student Member, IEEE*, Bohan Cui, *Graduate Student Member, IEEE*, Yiding Ji, *Member, IEEE*, and Xiang Yin, *Member, IEEE*

***Abstract*—Many property verification and enforcement problems of partially observed discrete event systems (DES) are typically addressed solely from the outside observer's perspective. However, systems may be monitored not only by a designated observer but also by a potentially malicious intruder when deployed in complex and open environments. The observer infers the information of the system based on its observation, which is referred to as *knowledge*. In addition, the intruder not only eavesdrops on the system's behaviors but also attempts to infer the observer's knowledge from its own observation. Notably, the information flow from the system to the observer and the intruder is *asymmetric*, resulting in incomparable observable events for each agent. Such an inference scenario is formalized as the *epistemic property*. This letter addresses the enforcement of the epistemic property in a partially observed DES through supervisory control. Specifically, we propose a game-theoretic framework to reflect the interaction between the observer/supervisor, the intruder, and the environment. A bipartite structure named all-protect structure is constructed as the game area, from which we solve the game and synthesize maximally permissive controllable and observable supervisors to provably enforce the properties.**

***Index Terms*—Discrete event systems, supervisory control, epistemic properties, asymmetric information, algorithmic games.**

Received 17 March 2025; revised 12 May 2025; accepted 2 June 2025. Date of publication 12 June 2025; date of current version 23 July 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62303389, Grant 62373289, and Grant 62173226; in part by the Guangdong Basic and Applied Basic Research Funding under Grant 2022A151511076 and Grant 2024A1515012586; in part by the Guangdong Scientific Research Platform and Project Scheme under Grant 2024KTSCX039; and in part by the Guangzhou-HKUST(GZ) Joint Funding Program under Grant 2024A03J0618 and Grant 024A03J0680. Recommended by Senior Editor A. P. Aguiar. *(Corresponding authors: Yiding Ji; Xiang Yin.)*

Shaowen Miao and Yiding Ji are with the Robotics and Autonomous Systems Thrust, Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China (e-mail: smiao585@connect.hkust-gz.edu.cn; jiyiding@hkust-gz.edu.cn).

Bohan Cui and Xiang Yin are with the School of Automation and Sensing, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: bohan_cui@sjtu.edu.cn; yinxiang@sjtu.edu.cn).

Digital Object Identifier 10.1109/LCSYS.2025.3578993

I. Introduction

PARTIALLY observed discrete event systems (DES) are ubiquitous in modeling high-level decision structures in cyber-physical systems, including computer and communication networks, automated manufacturing systems, and software systems. A substantial body of research has focused on various observational properties within DESs, encompassing topics such as observability [1], detectability [2], diagnosability [3], and opacity [4], among others [5], [6], [7], [8], [9], [10]. However, these studies typically investigate problems from a single agent, either a neutral observer or a malicious intruder.

Take opacity for example, supervisory control is commonly employed to address the opacity enforcement problem, where the intruder and supervisor/observer have incomparable observations, and the intruder is unaware of the supervisor's existence [11]. In contrast, [12] assumes that the intruder knows the existence of supervisors and introduces a nondeterministic control mechanism to enforce opacity, which enhances the system's plausible deniability. A uniform approach is proposed in [13] for the enforcement of various properties by supervisory control. The works [14], [15] design distinct obfuscation strategies based on insertion and edit functions to tackle the challenge of opacity enforcement. Recently, they have been extended to a modification function [16] to reduce synthesis complexity.

Meanwhile, recent works have studied decision making [17] and controller synthesis [18] in cooperative settings where multiple observational agents engage in knowledge reasoning. One critical challenge is that the systems are vulnerable to hostile environments where malicious intruders attempt to infer secrets and launch attacks [19], [20], [21], [22].

This letter considers a scenario in which the system is observed by both an observer and an intruder, characterized by asymmetric observation capabilities. The observer focuses on the system's direct behavior, inferring certain conclusions as its *knowledge* based on these observations. Conversely, the intruder endeavors to infer the observer's knowledge and the inference capacity is formally defined as *epistemic properties* in [23], which also proposes several approaches for verification of the properties. We extend the framework of [23] to study supervisory control for the enforcement of epistemic properties in a game-theoretic setting, which involves the supervisor,

the intruder, and the environment. A specialized game arena, termed the all-protect structure (APS), is constructed to derive maximally permissive controllable and observable supervisors.

The remainder of this letter is organized as follows. Section II reviews necessary preliminaries of the DES model and supervisory control. Section III formally defines the epistemic property and the corresponding enforcement problem. In Section IV, we propose algorithms to construct the game arena, then solve the game to synthesize maximally permissive supervisors that enforce the properties. Finally, Section V concludes this letter and discusses potential extensions.

II. Preliminaries

Notations and Automata Theory [24]. Let $\mathbb{N}$ denote the set of non-negative integers. For a set S, $|S|$ denotes its cardinality, and 2^S denotes its power set. An alphabet Σ is a finite, non-empty set of events. A string over Σ is a sequence of events from Σ, with the empty string denoted by ε. The set of all finite strings over Σ is denoted by Σ^*. A language L over Σ is a subset of Σ^*. For two strings $u, v \in \Sigma^*$, the *concatenation* of u and v is denoted by uv.

A *projection* $R\colon \Sigma^* \to \Sigma'^*$, with $\Sigma' \subseteq \Sigma$, is a morphism for concatenation defined as follows: $R(a) = \varepsilon$ for $a \in \Sigma \setminus \Sigma'$, $R(a) = a$ for $a \in \Sigma'$, and $R(a_1 a_2 \cdots a_n) = R(a_1)R(a_2)\cdots R(a_n)$ for $a_1 a_2 \cdots a_n \in \Sigma^*$. The inverse projection $R^{-1}\colon \Sigma'^* \to 2^{\Sigma^*}$ is defined as $R^{-1}(t) = \{s \in \Sigma^* \mid R(s) = t\}$. The definitions of the projection and its inverse can be extended to languages in the usual way.

A *deterministic finite automaton* (DFA) is a quadruple $G = (Q, \Sigma, \delta, q_0)$, where Q is a finite set of states, Σ is an alphabet, $q_0 \in Q$ is the initial state, and $\delta\colon Q \times \Sigma \to Q$ is the partial transition function that can be extended to $Q \times \Sigma^*$ by recursion. The language *generated* by G is $L(G) = \{s \in \Sigma^* \mid \delta(q_0, s) \in Q\}$. We denote $\Delta_G(q) = \{e \in \Sigma \mid \delta(q, e)\ !\}$ by the set of *active events* of G at state q, where ! means "is defined".

In DFA $G = (Q, \Sigma, \delta, q_0)$, for any subset $Q' \subseteq Q$, its unobservable reach with respect to the alphabet $\Sigma' \subseteq \Sigma$, denoted by $\mathrm{UR}_{\Sigma'}(Q')$, is defined as $\mathrm{UR}_{\Sigma'}(Q') = \{q \in Q \mid \exists q' \in Q', \exists s \in \Sigma'^*\colon q = \delta(q', s)\}$. The observer [24] of a DFA G, with respect to the alphabet $\Sigma' \subseteq \Sigma$, is defined as $\mathrm{Obs}_{\Sigma'}(G) = (X, \Sigma', f, x_0) = \mathrm{Ac}(2^Q, \Sigma', f, \mathrm{UR}_{\Sigma'}(\{q_0\}))$, where $\mathrm{Ac}(\cdot)$ denotes the accessible part and $f\colon X \times \Sigma' \to X$ is the deterministic transition function defined as $f(x, e) = \mathrm{UR}_{\Sigma'}(\bigcup_{q' \in x}\{\delta(q', e)\})$, which can be extended to $X \times \Sigma'^*$ by recursion. For any observation $w \in R(L(G))$, we denote by $\hat{Q}(w)$ the *current state estimate* based on w, defined as $\hat{Q}(w) = \{\delta(s) \in Q \mid \exists s \in L(G)\colon R(s) = w\}$.

The *parallel composition* of DFAs $G_i = (Q_i, \Sigma_i, \delta_i, q_i^0)$ for $i = 1, 2$, denoted by $G_1 \| G_2$, is the DFA $\mathrm{Ac}(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_{1,2}, (q_1^0, q_2^0))$, where $\delta_{1,2}((q_1, q_2), e) =$

$$\begin{cases} (\delta(q_1, e), \delta(q_2, e)), & \text{if } e \in \Delta_{G_1}(q_1) \cap \Delta_{G_2}(q_2); \\ (\delta(q_1, e), q_2), & \text{if } e \in \Delta_{G_1}(q_1) \setminus \Sigma_2; \\ (q_1, \delta(q_2, e)), & \text{if } e \in \Delta_{G_2}(q_2) \setminus \Sigma_1; \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

Supervisory Control [24]. For a partially observed and partially controlled DFA G, the alphabet Σ is partitioned into the set of *observable events* Σ_o and the set of *unobservable events* $\Sigma_{uo} = \Sigma \setminus \Sigma_o$, as well as the set of *controllable events* Σ_c and the set of *uncontrollable events* $\Sigma_{uc} = \Sigma \setminus \Sigma_c$.

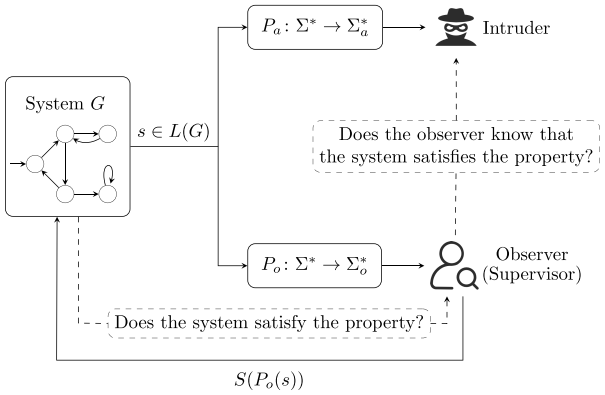

Fig. 1. Conceptual illustration of knowledge reasoning.

Let $\Gamma = \{\gamma \subseteq \Sigma \mid \Sigma_{uc} \subseteq \gamma\}$ be the set of control decisions and $P_o\colon \Sigma^* \to \Sigma_o^*$ be the supervisor's observation mapping of G. The *supervisor* of G with respect to Γ is defined by $S\colon P_o(L(G)) \to \Gamma$. Given G and S, the *closed-loop system* is denoted by S/G. The language generated by S/G is inductively defined by (i) $\varepsilon \in L(S/G)$ and (ii) $s \in L(S/G) \wedge sa \in L(G) \wedge a \in S(P_o(s)) \implies sa \in L(S/G)$.

A language $K \subseteq L(G)$ is *controllable* w.r.t. $L(G)$ and Σ_{uc} if $\overline{K}\Sigma_{uc} \cap L(G) \subseteq \overline{K}$. A language K is *observable* w.r.t. $L(G)$, $P_o\colon \Sigma^* \to \Sigma_o^*$, and Σ_c if $\forall s \in \overline{K}$, $\forall a \in \Sigma_c$, $sa \notin \overline{K} \wedge sa \in L(G) \implies P_o^{-1}[P_o(s)]a \cap \overline{K} = \emptyset$.

III. Problem Formulation

In this section, we extend the framework developed in [23] and formulate a property enforcing supervisory control problem, cf. Fig. 1. Specifically, we refer to the conclusion drawn by an observer based on its own observation as "knowledge", such as the statement "the observer knows that the system is opaque". In this scenario, there is a passive intruder who can not only observe the system independently but also infer the observer's knowledge from its own observations.

We denote the observable events of the observer by Σ_o and the unobservable events by $\Sigma_{uo} = \Sigma \setminus \Sigma_o$, while the observable and unobservable events of the intruder are denoted by Σ_a and $\Sigma_{ua} = \Sigma \setminus \Sigma_a$, respectively. The partial observation properties of the observer and the intruder are modeled by $P_o\colon \Sigma^* \to \Sigma_o^*$ and $P_a\colon \Sigma^* \to \Sigma_a^*$, respectively.

Formally, the *knowledge* of the observer can be defined as a predicate: $\mathrm{Kw}_o\colon P_o(L(G)) \to \{\mathrm{T}, \mathrm{F}\}$. For any $w \in P_o(L(G))$, $\mathrm{Kw}_o(w) = \mathrm{T}$ indicates that some knowledge of interest holds based on the observation w. The concrete expression of Kw_o depends on the context of the problem being studied and will be specified later. In addition, the *knowledge estimate* of the intruder is defined as: $\widehat{\mathrm{Kw}}_{ao}\colon P_a(L(G)) \to \{\mathrm{Y}, \mathrm{N}, \mathrm{U}\}$ such that for any $s \in P_a(L(G))$,

$$\widehat{\mathrm{Kw}}_{ao}(s) = \begin{cases} \mathrm{Y}, \text{ if } \big(\forall w \in P_o\big(P_a^{-1}(s) \cap L(G)\big)\big)[\mathrm{Kw}_o(w) = \mathrm{T}]; \\ \mathrm{N}, \text{ if } \big(\forall w \in P_o\big(P_a^{-1}(s) \cap L(G)\big)\big)[\mathrm{Kw}_o(w) = \mathrm{F}]; \\ \mathrm{U}, \text{ otherwise.} \end{cases} \tag{1}$$

In words, the knowledge estimate returns Y (resp. N) if the intruder can deduce that the observer has determined that some

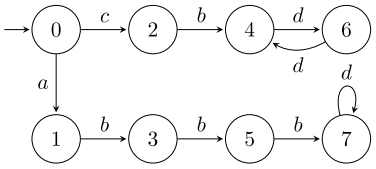

Fig. 2. DFA G with $\Sigma_o = \{b, d\}$, $\Sigma_a = \{a, b\}$, and $\Sigma_c = \{b\}$.

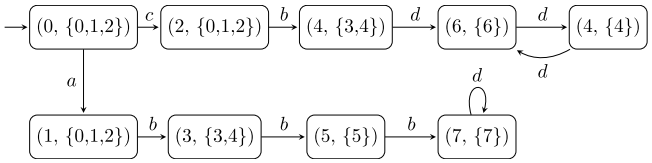

Fig. 3. Knowledge recognizer $\tilde{G}$.

knowledge is true (resp. false), and returns U if the intruder is uncertain about it. Then we recall the *epistemic property* introduced in [23], which is a general definition that will be specified later, depending on the expression of Kw_o.

Definition 1 (Epistemic Property): Given a system G, the projection of the observer P_o, the projection of the intruder P_a, the knowledge Kw_o, and the knowledge estimate $\widehat{\mathrm{Kw}}_{ao}$, the *epistemic property* is defined as:

$$(\mathbb{Q}s \in L(G)\colon \mathrm{Kw}_o(P_o(s)) = \mathbb{K}_o)\big[\widehat{\mathrm{Kw}}_{ao}(P_a(s)) = \mathbb{K}_{ao}\big],$$

where $\mathbb{Q} \in \{\exists, \forall\}$, $\mathbb{K}_o \in \{\mathrm{T}, \mathrm{F}\}$, and $\mathbb{K}_{ao} \in \{\mathrm{Y}, \mathrm{N}, \mathrm{U}\}$.

By definition, a tuple $\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao}\rangle$ indicates a specific epistemic property. We write $G \models \langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao}\rangle$ if G satisfies the epistemic property defined by these parameters.

In [23], eight types of epistemic properties are specified, including high-order opacity, intrusion undetectability, epistemic diagnosability, and others. For simplicity, we use high-order opacity as a typical example for the remainder of this letter. However, we will demonstrate in the next section that the proposed enforcement approach in this letter is generally applicable to all eight epistemic properties.

Definition 2 (High-Order Opacity): Let the specification be $T_{spec} = \{(q, q') \in Q \times Q \mid q \neq q'\}$, representing the set of state pairs to be distinguished, and the knowledge be defined as $\mathrm{Kw}_o(w) = \mathrm{T} \iff [\hat{Q}(w) \times \hat{Q}(w)] \cap T_{spec} = \emptyset$ for $w \in P_o(L(G))$. A system G is *high-order opaque* w.r.t. the projections P_o, P_a, and the knowledge Kw_o, if

$$(\forall s \in L(G))\big[\mathrm{Kw}_o(P_o(s)) = \mathrm{T} \implies$$
$$(\exists t \in L(G))[\mathrm{Kw}_o(P_o(t)) = \mathrm{F} \wedge P_a(s) = P_a(t)]\big].$$

It is straightforward to see that a system G is high-order opaque if and only if $G \models \langle \forall, \mathrm{Kw}_o, \mathrm{T}, \mathrm{U}\rangle$ with $\mathrm{Kw}_o(w) = \mathrm{T} \iff [\hat{Q}(w) \times \hat{Q}(w)] \cap T_{spec} = \emptyset$ for $w \in P_o(L(G))$. By specifying different T_{spec}, the high-order opacity can also have different physical meanings, see [23] for details.

Example 1: Consider the system G depicted in Fig. 2, it is not high-order opaque by Definition 2. Since for $s = abbbd^n$ with $P_o(s) = bbbd^n$, $n \in \mathbb{N}$, $\hat{Q}(bbbd^n) = \{7\}$ and $\mathrm{Kw}_o(bbbd^n) = \mathrm{T}$, there are no string t such that $\mathrm{Kw}_o(P_o(t)) = \mathrm{F}$ and $P_a(t) = P_a(s) = abbb$. This means that for the string s, the observer observes $bbbd^n$ and concludes that "the system's current state is unique", while the intruder infers this knowledge based on its own observation of $abbb$.

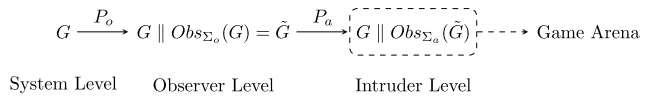

Fig. 4. Game arena construction flow chart.

If the system does not satisfy a given epistemic property, a supervisor is synthesized to restrict the system's behaviors to enforce this property, and the supervisor should be as permissive as possible. Then we formulate the key problem of this letter.

Problem 1: Given a system G, a projection of the observer P_o, and an epistemic property $\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao}\rangle$, synthesis a supervisor $S\colon P_o(L(G)) \to \Gamma$ such that:

1) $S/G \models \langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao}\rangle$;
2) $L(S/G)$ is controllable w.r.t. $L(G)$ and Σ_{uc} and observable w.r.t. $L(G)$, $P_o\colon \Sigma^* \to \Sigma_o^*$, and Σ_c;
3) For any other supervisor S' satisfying the above two conditions, $L(S/G) \not\subset L(S'/G)$ holds.

In this letter, we assume that the intruder does not know the existence of supervisors. Consequently, when substituting S/G in Condition 1) into Definition 1, the formula for $\widehat{\mathrm{Kw}}_{ao}$ is exactly in the form of Eq. (1). However, if the intruder knows the supervisor, the intersection with $L(G)$ in Eq. (1) should be replaced by the intersection with $L(S/G)$ instead. This means that the intruder can update its observations based on the supervisor's decisions, which raises a more complex scenario to be explored in our future work. For a more detailed discussion on the differences between scenarios where the intruder knows or does not know the supervisor (control policy), please refer to the works [11], [14].

IV. Supervisory Control for Epistemic Properties

A. General Game Arena

In this subsection, we first construct the knowledge recognizer, followed by the double estimator, and then proceed to define the game arena for supervisor synthesis. The overall construction flow chart of the game arena is shown in Fig. 4.

The *knowledge recognizer* is used for estimating the system state, which essentially captures the knowledge of the observer.

Definition 3 (Knowledge Recognizer): Given a DFA G and an alphabet Σ_o of the observer, the knowledge recognizer of G is defined as $\tilde{G} = G\|Obs_{\Sigma_o}(G) = (\tilde{Q}, \Sigma, \tilde{\delta}, \tilde{q}_0)$.

Within the knowledge recognizer $\tilde{G}$, we define the set of *known states* as $\tilde{Q}_{\mathrm{T}} = \{\tilde{q} = (q, x) \in \tilde{Q} \mid \exists w \in P_o(L(G))\colon f(x_0, w) = x \wedge \mathrm{Kw}_o(w) = \mathrm{T}\}$.

Next, we define the *double estimator* for the knowledge estimation on the observer from the intruder's perspective.

Definition 4 (Double Estimator): Given a DFA G, an alphabet Σ_o of the observer, an alphabet Σ_a of the intruder, the double estimator of G is the observer of $\tilde{G}$ with respect to Σ_a, denoted by $Obs_{\Sigma_a}(\tilde{G}) = (\tilde{X}, \Sigma_a, \tilde{f}, \tilde{x}_0)$.

We now repeat two of the eight criteria in [23, Th. 1] to verify the epistemic properties.

Lemma 1 (Criterion for Checking Epistemic Property):

1) $G \models \langle \forall, \mathrm{Kw}_o, \mathrm{T}, \mathrm{U}\rangle \iff \forall \tilde{x} \in \tilde{X}\colon \tilde{x} \not\subseteq \tilde{Q}_{\mathrm{T}}$;
2) $G \models \langle \exists, \mathrm{Kw}_o, \mathrm{T}, \mathrm{Y}\rangle \iff \exists \tilde{x} \in \tilde{X}\colon \tilde{x} \subseteq \tilde{Q}_{\mathrm{T}}$;

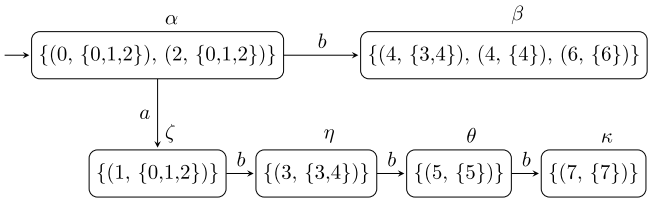

Fig. 5. Double estimator $Obs_{\Sigma_a}(\tilde{G})$.

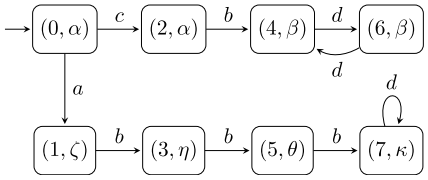

Fig. 6. Parallel automaton $G \| Obs_{\Sigma_a}(\tilde{G})$.

For an epistemic property $\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao} \rangle$, we denote the second part (after the colon) of its establishment condition by $\Phi(\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao} \rangle)$, e.g., $\Phi(\langle \forall, \mathrm{Kw}_o, \mathrm{T}, \mathrm{U} \rangle) = \tilde{x} \not\subseteq \tilde{Q}_{\mathrm{T}}$.

Example 2: Reconsider the system G depicted in Fig. 2; its knowledge recognizer $\tilde{G}$ is illustrated in Fig. 3; and its double estimator $Obs_{\Sigma_a}(\tilde{G})$ is shown in Fig. 5. Taking high-order opacity as an example, we let $T_{spec} = \{(q, q') \in Q \times Q \mid q \neq q'\}$, and define $\mathrm{Kw}_o(w) = \mathrm{T} \iff [\hat{Q}(w) \times \hat{Q}(w)] \cap T_{spec} = \emptyset$ for $w \in P_o(L(G))$. We then derive that the set of known states is $\tilde{Q}_{\mathrm{T}} = \{(4, \{4\}), (6, \{6\}), (7, \{7\})\}$. Furthermore, since the state $\{(7, \{7\})\}$ in $Obs_{\Sigma_a}(\tilde{G})$ satisfies $\{(7, \{7\})\} \subseteq \tilde{Q}_{\mathrm{T}}$, by condition 1) of Lemma 1, we conclude that $G \not\models \langle \forall, \mathrm{Kw}_o, \mathrm{T}, \mathrm{U} \rangle$, which indicates that the system is not high-order opaque.

We then construct $G \| Obs_{\Sigma_a}(\tilde{G}) = (X_p, \Sigma, f_p, (q_0, \tilde{x}_0))$ to track both the system's behaviors and the knowledge estimation. Based on $G \| Obs_{\Sigma_a}(\tilde{G})$, we define the game arena, which incorporates the supervisor's decisions, the intruder's knowledge estimation, and the environmental confrontation.

Definition 5 (Game Arena): A game arena w.r.t. system $G = (Q, \Sigma, \delta, q_0)$ and control decisions $\Gamma = \{\gamma \subseteq \Sigma \mid \Sigma_{uc} \subseteq \gamma\}$, is defined as $A = (Y_D, Y_E, \xi_{DE}, \xi_{ED}, \Sigma, \Gamma, y_d^0)$, where

- $Y_D \subseteq 2^{X_p}$ is the set of *decision states* (D-states);
- $Y_E \subseteq 2^{X_p} \times \Gamma$ is the set of *environment states* (E-states), and for any $y_e \in Y_E$, we write $y_e = (\mathcal{S}(y_e), \Gamma(y_e))$, where $\mathcal{S}(y_e)$ and $\Gamma(y_e)$ denote the state and control decision components of y_e, respectively;
- $\xi_{DE} \colon Y_D \times \Gamma \to Y_E$ is the partial transition function from D-states to E-states, satisfying the condition:

$$\xi_{DE}(y_d, \gamma) = y_e \implies \big[\mathcal{S}(y_e) = \mathrm{UR}_{(\Sigma_{uo} \cap \gamma)}(\{y_d\})\big] \wedge \big[\Gamma(y_e) = \gamma\big]; \quad (2)$$

- $\xi_{ED} \colon Y_E \times \Sigma \to Y_D$ is the partial transition function from E-states to D-states, satisfying the condition:

$$\xi_{ED}(y_e, e) = y_d \implies \big[e \in \Gamma(y_e) \cap \Sigma_o\big] \wedge \big[y_d = \cup_{x_p \in \mathcal{S}(y_e)} \{f_p(x_p, e) \mid f_p(x_p, e)!\}\big]; \quad (3)$$

- $y_d^0 = \{(q_0, \tilde{x}_0)\} \in Y_D$ is the initial D-state;

In a game arena A, the supervisor plays against the intruder and the environment. A transition from a D-state y_d to an E-state y_e represents the control decisions issued by the supervisor. The supervisor plays at D-states by issuing control decisions following the knowledge estimation from the intruder. At this point, the supervisor updates the current possible states in $G \| Obs_{\Sigma_a}(\tilde{G})$ at the next E-state y_e, i.e., $\mathcal{S}(y_e) = \mathrm{UR}_{(\Sigma_{uo} \cap \gamma)}(\{y_d\})$. The environment plays at E-states by choosing to fire events from the set of latest enabled events γ, recorded in the right component of the E-state y_e, i.e., $\Gamma(y_e) = \gamma$. A transition from y_e to y_d represents the observable reach in one step. Thus, it holds that $y_d = \bigcup_{x_p \in \mathcal{S}(y_e)} \{f_p(x_p, e) \mid f_p(x_p, e)!\}$, where $e \in \Gamma(y_e) \cap \Sigma_o$.

We also define the set of control decisions at a D-state $y_d \in Y_D$ as $C_A(y_d) = \{\gamma \in \Gamma \mid \xi_{DE}(y_d, \gamma)!\}$. The following requirements are imposed to induce feasible supervisors:

(i) for any $y_d \in Y_D$, $C_A(y_d) \neq \emptyset$;

(ii) for any $y_e \in Y_E$, $\forall e \in \Gamma(y_e) \cap \Sigma_o \colon (\exists x_p \in \mathcal{S}(y_e) \colon f_p(x_p, e)!) \implies \xi_{ED}(y_e, e)!$.

The first condition requires that there always exists a control decision at any D-state, and the second requires that any enabled active observable events should be allowed to occur. A game arena is called *complete* if it satisfies both conditions.

Algorithm 1: Construct the APS

Input: G, Γ, and $\mathrm{EP} = \langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao} \rangle$
Output: $APS(G)$
$\mathbf{Y_D} \leftarrow \{y_d^0\}$ and $\mathbf{Y_E} \leftarrow \emptyset$;
DoDFS($G, y_d^0, APS(G)$, EP);
while *there are D-states without successors* **do**
  Remove such D-states and their predecessor E-states;
$APS(G) \leftarrow \mathrm{Ac}(APS(G))$;
Procedure: DoDFS($G, y_d, APS(G)$, EP)
for $\gamma \in \Gamma$ **do**
  $y_e \leftarrow \xi_{DE}(y_d, \gamma)$;
  if $\mathbb{Q}x_p = (q, \tilde{x}) \in \mathcal{S}(y_e) \colon \Phi(\mathrm{EP})$ **then**
    Add transition $y_d \xrightarrow{\gamma} y_e$ to $\Xi_{\mathbf{DE}}$;
    if $y_e \notin \mathbf{Y_E}$ **then**
      $\mathbf{Y_E} \leftarrow \mathbf{Y_E} \cup \{y_e\}$;
      for $e \in \gamma \cap \Sigma_o$ **do**
        $y_d' \leftarrow \xi_{ED}(y_e, e)$;
        if $y_d' \neq \emptyset$ **then**
          Add transition $y_e \xrightarrow{e} y_d'$ to $\Xi_{\mathbf{ED}}$;
          if $y_d' \notin \mathbf{Y_D}$ **then**
            $\mathbf{Y_D} \leftarrow \mathbf{Y_D} \cup \{y_d'\}$;
            DoDFS($G, y_d', APS(G)$, EP);

B. All-Protect Structure

In this subsection, we first construct a special game arena to include all supervisors protecting the observer's knowledge, namely, *all-protect structure* (APS). Then, we show how to synthesize a maximally permissive supervisor from the APS.

Definition 6 (All-Protect Structure): The APS w.r.t. system G and epistemic property $\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao} \rangle$, denoted by $APS(G) = (\mathbf{Y_D}, \mathbf{Y_E}, \Xi_{\mathbf{DE}}, \Xi_{\mathbf{ED}}, \Sigma, \Gamma, y_d^0)$, is defined as the *largest* and *complete* game arena w.r.t. G and Γ, such that: 1) it is complete; 2) for any $y_e \in \mathbf{Y_E}$, it holds that $\mathbb{Q}x_p = (q, \tilde{x}) \in \mathcal{S}(y_e) \colon \Phi(\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao} \rangle)$.

Being the "largest" implies that the APS will violate at least one condition if additional D- or E-states are included to the structure. Condition 2) corresponds to the verification criterion

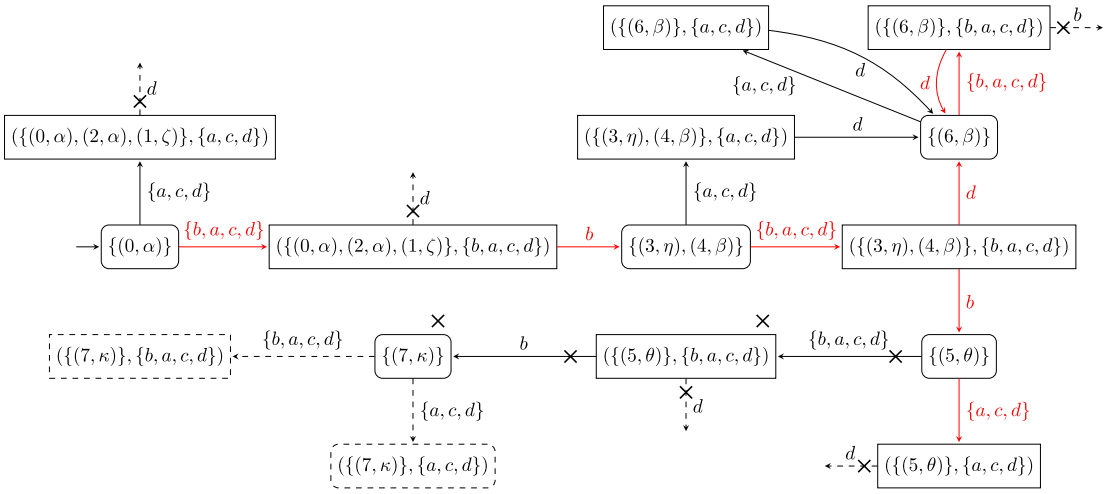

Fig. 7. *APS*(G) constructed by Algorithm 1, where dashed boxes, dashed lines, boxes with crosses, and lines with crosses are excluded. The rounded rectangle represents the D-state, and the right-angled rectangle represents the E-state. Each dashed line with a slash points to an empty set D-state.

in Lemma 1, which ensures that the APS satisfies the given epistemic property.

Algorithm 1 constructs the APS in two steps. First, we develop the game arena by performing a depth-first search (DFS) in Lines 6–18, which enumerates all possible control decisions for each state and includes all states satisfying the given epistemic property. Then, we iteratively prune the states that violate the completeness condition from the remaining part of the arena (Lines 3–4) until the structure converges.

Example 3: Let us continue to consider the high-order opacity and let the knowledge be $G \models \langle \forall, \mathrm{Kw}_o, \mathrm{T}, \mathrm{U} \rangle$ with $\mathrm{Kw}_o(w) = \mathrm{T} \iff [\hat{Q}(w) \times \hat{Q}(w)] \cap T_{spec} = \emptyset$. The all-protect structure *APS*(G) is illustrated in Fig. 7, which is constructed based on the parallel automaton $G \| Obs_{\Sigma_a}(\tilde{G})$, as shown in Fig. 6. At the initial D-state $y_d^0 = \{(0, \alpha)\}$, the supervisor can choose either the control decisions $\{a, c, d\}$ or $\{b, a, c, d\}$. The DFS procedure stops at the D-state $\{(7, \kappa)\}$, which will be removed in Lines 3–4. This is because, both E-states $(\{(7, \kappa)\}, \{b, a, c, d\})$ and $(\{(7, \kappa)\}, \{a, c, d\})$ will not be included in *APS*(G) according to Line 8, as they violate the high-order opacity by Lemma 1. This results in the D-state $\{(7, \kappa)\}$ having no successor E-states. Notably, in addition to $\{(7, \kappa)\}$, its predecessor E-state $(\{(5, \theta)\}, \{b, a, c, d\})$ will also be removed in Lines 3–4. This means that the control decision $\{b, a, c, d\}$ at state 5 of G is not allowed.

Algorithm 2 synthesizes a maximally permissive supervisor from *APS*(G), which enforces the given epistemic property. This algorithm performs a DFS (Lines 5–14) and chooses one locally maximal control decision at each D-state (Line 5).

Theorem 2: Given a system G and an epistemic property $\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao} \rangle$, the closed-loop system S/G satisfies the epistemic property, where the supervisor S is designed using Algorithm 2, which corresponds to the solution to Problem 1.

Proof: First, the game arena in Definition 5 is constructed from $G \| Obs_{\Sigma_a}(\tilde{G})$, which simultaneously tracks both the behavior of the system G and the estimation of knowledge from the perspective of the intruder. By the definition of $Obs_{\Sigma_a}(\tilde{G})$ and the property of parallel composition, we have $L(G) = L(G \| Obs_{\Sigma_a}(\tilde{G}))$. Next, Algorithm 1 correctly constructs the APS. Specifically, Line 6 ensures that the DFS traverses all feasible control decisions, and Line 11 ensures that the DFS explores all enabled and active observable events which are included through ξ_{DE} defined in Eq. (2). Thus, no additional states are to be included, and the APS is the *largest*. Then, the *completeness* of the APS is assured by Lines 3–4 and 12. Furthermore, the APS enforces the given epistemic property, as guaranteed by Lemma 1 and Line 8. Given that $\Gamma = \{\gamma \subseteq \Sigma \mid \Sigma_{uc} \subseteq \gamma\}$, the *controllability* of $L(S/G)$ is ensured by Line 6. *Observability* is ensured by Line 11 and Eq. (2). Finally, Algorithm 2 traverses the constructed APS and selects a locally maximal control decision at each D-state, thereby ensuring the *maximal permissiveness* of $L(S/G)$. ■

When $\Sigma_c \subseteq \Sigma_o$, it is well known that observability and controllability jointly imply normality [24], thus there exists a unique controllable and observable supervisor that enforces the epistemic property. This result leads to the following corollary of supervisor synthesis.

Algorithm 2: Synthesis Supervisor

Input: *APS*(G)
Output: S
1 Mark y_d^0 as "visited";
2 $w \leftarrow \varepsilon$;
3 Traverse(*APS*(G), w, y_d^0);
4 **return** S

Procedure: Traverse(*APS*(G), w, y_d)
5 Choose a locally maximal control decision $\gamma \in C_{APS(G)}(y_d)$ s.t. $\forall \gamma' \in C_{APS(G)}(y_d) : \gamma \not\subset \gamma'$;
6 $S(w) \leftarrow \gamma$; $y_e \leftarrow \Xi_{\mathbf{DE}}(y_d, \gamma)$;
7 **if** y_e *is not marked as "visited"* **then**
8 Mark y_e as "visited";
9 **for** $e \in \gamma \cap \Sigma_o$ **do**
10 $w \leftarrow we$;
11 $y'_d \leftarrow \Xi_{\mathbf{ED}}(y_e, e)$;
12 **if** y'_d *is not marked as "visited"* **then**
13 Mark y'_d as "visited";
14 Traverse(*APS*(G), w, y'_d);

Corollary 3: Given a system G and an epistemic property $\langle \mathbb{Q}, \mathrm{Kw}_o, \mathbb{K}_o, \mathbb{K}_{ao} \rangle$, assume that $\Sigma_c \subseteq \Sigma_o$ ($\Sigma_{uo} \subseteq \Sigma_{uc}$), Algorithm 2 returns a unique supervisor S such that the closed-loop system S/G satisfies the epistemic property.

Proof: We first prove the uniqueness of the output of Algorithm 2. By substituting $\Sigma_{uo} \subseteq \Sigma_{uc}$ and $\Gamma = \{\gamma \subseteq \Sigma \mid \Sigma_{uc} \subseteq \gamma\}$ into Eq. (2), we can rewrite Eq. (2) as $\xi_{DE}(y_d, \gamma) = y_e \Rightarrow [\mathcal{S}(y_e) = \mathrm{UR}_{\Sigma_{uo}}(\{y_d\})] \wedge [\Gamma(y_e) = \gamma]$. This implies that the state component $\mathcal{S}(y_e)$, also known as the current state estimate, of the reached E-state y_e does not depend on the selected control decision. From Eq. (3), we know that $e \in \Gamma(y_e) \cap \Sigma_o$. Given that $\Sigma_c \subseteq \Sigma_o$ and $\Gamma = \{\gamma \subseteq \Sigma \mid \Sigma_{uc} \subseteq \gamma\}$, we deduce that $\Gamma(y_e) = \Sigma_{uc} \cup \Sigma'_o$ for some $\Sigma'_o \subseteq \Delta(\mathcal{S}(y_e))$, where $\Delta(\mathcal{S}(y_e)) = \bigcup_{x_\| \in \mathcal{S}(y_e)} \Delta_{G\|Obs_{\Sigma_a}(\tilde{G})}(x_\|)$. Since the control decision does not affect $\mathcal{S}(y_e)$, $\Delta(\mathcal{S}(y_e))$ remains unchanged. This means that the larger the control decision $\Gamma(y_e)$ we choose, the more the system's subsequent behavior will be. Therefore, within $APS(G)$, we can always choose $\Gamma(y_e) = \Sigma_{uc} \cup \Delta(\mathcal{S}(y_e))$. Theorem 2 guarantees that all supervisors induced from $APS(G)$ can correctly enforce the given epistemic property, which completes the proof. ■

Example 4: Consider again $APS(G)$ depicted in Fig. 7. The traversed paths are highlighted in red, which indicates the supervisor S. Specifically, $S(\varepsilon) = S(b) = S(bd^n) = \{b, a, c, d\}$ and $S(bb) = \{a, c, d\}$. Therefore, the closed-loop system S/G is obtained from G shown in Fig. 2 by removing the state 7 as well as transitions $5 \xrightarrow{b} 7$ and $7 \xrightarrow{d} 7$.

Note that since liveness is not involved in the definition of epistemic property for the original system G, we do not require the closed-loop system to be live. As a result, the supervisor designed using our approach may result in a blocking closed-loop system. This issue is easily mitigated by leveraging existing approaches such as [13], [25] to enforce liveness.

Remark 4: The DFS procedure of Algorithm 1 constructs a game arena with $2^{|X_p|+|\Sigma_c|}$ states, where $|X_p|$ is doubly exponential in the size of G. The complexity of Lines 3–4 is quadratic in the size of the constructed game arena. Finally, Algorithm 2 operates with linear complexity in the size of $APS(G)$. As a result, the overall synthesis procedure has triply exponential complexity in the size of G. We argue that, although some related works [11], [12], [13] exhibit lower synthesis complexity, they address relatively tractable properties. Specifically, the synthesis procedure in [13] has a singly exponential complexity; however, it does not account for the intruder's perspective. Subsequent works [11], [12], do incorporate the intruder's perspective, but their synthesis complexity remains doubly exponential. Nonetheless, these two works do not consider the scenario that the intruder may be interested in inferring the observer's *knowledge*. On the other hand, following a similar synthesis procedure as in this letter, the twin estimator and state pair estimator in [23] can be readily adapted to address some of the eight epistemic properties, with a predictably doubly exponential complexity.

V. Conclusion

This letter studies the enforcement of epistemic properties in DES, which aims to prevent malicious intruders from inferring the observer's knowledge. By integrating observer state estimation, intruder knowledge estimation, and system-environment interaction, we propose a game-theoretic method to synthesize property-enforcing and maximally permissive supervisors. Future work will extend the framework to more complicated settings where intruders update their observations when events are dynamically disabled.

References

[1] Y. Hou, Q. Li, Y. Ji, G. Wang, and C.-Y. Weng, "A new approach for verification of delay co-observability of discrete-event systems," *IEEE Trans. Control Netw. Syst.*, vol. 10, no. 3, pp. 1542–1554, Sep. 2023.

[2] L. Zhou, S. Shu, and F. Lin, "Detectability of discrete-event systems under nondeterministic observations," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 3, pp. 1315–1327, Jul. 2021.

[3] F. Lin, S. Lafortune, and C. Wang, "Diagnosability and attack detection for discrete event systems under sensor attacks," *Discr. Event Dyn. Syst.*, vol. 34, no. 3, pp. 465–495, 2024.

[4] S. Miao, A. Lai, and J. Komenda, "Always guarding you: Strong initial-and-final-state opacity of discrete-event systems," *Automatica*, vol. 173, Mar. 2025, Art. no. 112085.

[5] M. V. Alves, A. E. da Cunha, L. K. Carvalho, M. V. Moreira, and J. C. Basilio, "Robust supervisory control of discrete event systems against intermittent loss of observations," *Int. J. Control*, vol. 94, no. 7, pp. 2008–2020, 2021.

[6] R. Tai, L. Lin, and R. Su, "Synthesis of optimal covert sensor–actuator attackers for discrete-event systems," *Automatica*, vol. 151, May 2023, Art. no. 110910.

[7] W. Fokkink and M. Goorden, "Offline supervisory control synthesis: Taxonomy and recent developments," *Discr. Event Dyna. Syst.*, vol. 34, pp. 605–657, Nov. 2024.

[8] R. Oura, T. Ushio, and A. Sakakibara, "Bounded synthesis and reinforcement learning of supervisors for stochastic discrete event systems with LTL specifications," *IEEE Trans. Autom. Control*, vol. 69, no. 10, pp. 6668–6683, Oct. 2024.

[9] R. Malik, S. Mohajerani, and M. Fabian, "A survey on compositional algorithms for verification and synthesis in supervisory control," *Discr. Event Dyn. Syst.*, vol. 33, no. 3, pp. 279–340, 2023.

[10] W. Deng, D. Qiu, and J. Yang, "Intersection-based decentralized supervisory control of probabilistic discrete event systems," *IEEE Trans. Autom. Control*, vol. 66, no. 12, pp. 6171–6178, Dec. 2021.

[11] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Current-state opacity enforcement in discrete event systems under incomparable observations," *Discr. Event Dyn. Syst.*, vol. 28, pp. 161–182, Jun. 2018.

[12] Y. Xie, X. Yin, and S. Li, "Opacity enforcing supervisory control using nondeterministic supervisors," *IEEE Trans. Autom. Control*, vol. 67, no. 12, pp. 6567–6582, Dec. 2022.

[13] X. Yin and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems," *IEEE Trans. Autom. Control*, vol. 61, no. 8, pp. 2140–2154, Aug. 2016.

[14] R. Liu, J. Lu, Y. Liu, X. Yin, and C. N. Hadjicostis, "Opacity enforcement via greedy privately-and-publicly known insertion functions," *IEEE Trans. Autom. Control*, vol. 69, no. 4, pp. 2500–2506, Apr. 2024.

[15] C. Keroglou and S. Lafortune, "Embedded insertion functions for opacity enforcement," *IEEE Trans. Autom. Control*, vol. 66, no. 9, pp. 4184–4191, Sep. 2021.

[16] X. Li, C. N. Hadjicostis, and Z. Li, "Opacity enforcement in discrete event systems using modification functions," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 3252–3264, 2024.

[17] K. Ritsuka and K. Rudie, "Do what you know: Coupling knowledge with action in discrete-event systems," *Discr. Event Dyn. Syst.*, vol. 33, no. 3, pp. 257–277, 2023.

[18] S. Takai and R. Kumar, "Nonexistence of upper bound to inferencing level in decentralized discrete event control," *IEEE Trans. Autom. Control*, vol. 69, no. 11, pp. 7964–7971, Nov. 2024.

[19] Y. Wang and M. Pajic, "Supervisory control of discrete event systems in the presence of sensor and actuator attacks," in *Proc. 58th IEEE Conf. Decis. Control*, 2019, pp. 5350–5355.

[20] R. Meira-Góes, S. Lafortune, and H. Marchand, "Synthesis of supervisors robust against sensor deception attacks," *IEEE Trans. Autom. Control*, vol. 66, no. 10, pp. 4990–4997, Oct. 2021.

[21] Y. Ji, X. Yin, and S. Lafortune, "Enforcing opacity by insertion functions under multiple energy constraints," *Automatica*, vol. 108, Oct. 2019, Art. no. 108476.

[22] Y. Ji, X. Yin, and S. Lafortune, "Opacity enforcement using nondeterministic publicly known edit functions," *IEEE Trans. Autom. Control*, vol. 64, no. 10, pp. 4369–4376, Oct. 2019.

[23] B. Cui, Z. Ma, S. Li, and X. Yin, "On epistemic properties in discrete-event systems: A uniform framework and its applications," 2024, *arXiv:2409.06588*.

[24] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 3rd ed. Cham, Switzerland: Springer, 2021.

[25] Y. Hu, Z. Ma, and Z. Li, "Design of supervisors for active diagnosis in discrete event systems," *IEEE Trans. Autom. Control*, vol. 65, no. 12, pp. 5159–5172, Dec. 2020.