Synthesis of Dynamic Masks for Infinite-Step Opacity

Xiang Yin, *Member, IEEE*, and Shaoyuan Li, *Senior Member, IEEE*

***Abstract*—We investigate the problem of synthesizing *dynamic masks* that preserve the *infinite-step opacity* in the context of discrete-event systems. Dynamic mask is an information acquisition mechanism that controls the observability of the system's events dynamically online, e.g., by turning sensors on/off. A system equipped with a dynamic mask is said to be infinite-step opaque if an outside intruder that can access all acquired information can never infer that the system was at some secret state for any specific previous instant. Existing works on the dynamic mask synthesis problem can only preserve the current-state opacity. However, synthesizing dynamic masks for the infinite-step opacity, which is stronger than the current-state opacity, is much more challenging. The main reason is that the delayed information is involved in this problem and whether or not a current secret can be revealed depends on sensing decisions to be synthesized in the future. In this paper, a new type of information state is proposed to capture all the delayed information in the infinite-step opacity synthesis problem. An effective algorithm is then presented to solve the synthesis problem, which extends existing dynamic mask synthesis techniques from the current-state opacity to infinite-step opacity. Additionally, an information-state-reduction-based approach is proposed to further mitigate the computational complexity of the synthesis procedure. Finally, we discuss how to generalize our results to a class properties with delayed information including infinite-step K-anonymity and infinite-step indistinguishability.**

***Index Terms*—Delayed information, discrete-event systems (DESs), dynamic masks, infinite-step opacity.**

I. Introduction

SECURITY and privacy are increasingly important issues in the analysis of networked cyber-physical systems. In this paper, we investigate an information-flow security property called opacity in the context of discrete-event systems (DESs), an important class of man-made cyber-physical systems with discrete state-spaces and event-triggered dynamics [7]. Roughly speaking, opacity is a confidentiality property that captures the *plausible deniability* of the system's "secret behavior." In other words, an opaque system should be secure in the sense that its secret should never be revealed to a passive observer (intruder) that is potentially malicious.

In the context of a DES, opacity has drawn considerable attention in the past decade; see, e.g., [1], [5], [17], [18], [24]–[26]. In particular, different notions of opacity were proposed in the literature in order to capture different secret requirements. This includes, for example, the current-state opacity [41], initial-state opacity [32], K-step opacity [29], and infinite-step opacity [31], [46]. Verifications of different notions of opacity have also been investigated in different system models. For example, Basile *et al.* [2], [6], [12], [36], [37] investigated how to verify different notions of opacity for the DES modeled by Petri nets. In [3], [11], and [21], the verification of opacity in a stochastic DES was investigated. Opacity has also been studied in infinite-state systems modeled by recursive tile systems [10] and pushdown systems [23]. More recently, Noori-Hosseini *et al.* [27], [51] investigated how to reduce the verification complexity of opacity using abstraction-based approaches. The reader is referred to [18] and [24] for a more comprehensive literature review.

Among many different notions of opacity, one most simple version is the current-state opacity. Specifically, a system is said to be current-state opaque if the intruder cannot determine for sure that the system is *currently* at a secret state based on its limited observation. Therefore, whether or not a system is opaque purely depends on the current information of the system. However, in many situations, even if the intruder cannot determine the secret of the system currently, it can keep observing the behavior of the system and use additional future information to improve its knowledge of the system in the past. This is essentially an information *smoothing* process. Therefore, even if a system is current-state opaque, the intruder may still be able to know that it was at a secret state after some delay. In order to capture this issue, infinite-step opacity was proposed by requiring that the intruder should never know for sure that the system was/is at a secret state for any specific instant. Clearly, infinite-step opacity is stronger than current-state opacity. Verification algorithms for infinite-step opacity have been provided in [31] and [46] for systems modeled as finite-state automata. More recently, infinite-step opacity has also been studied in the context of a stochastic DES [49].

When a system is verified to be nonopaque, another important problem in opacity is to *synthesize* an opaque system. This problem has recently been widely studied in the literature and several different synthesis/enforcement mechanisms have been

Manuscript received August 24, 2018; revised March 20, 2019; accepted May 11, 2019. Date of publication May 15, 2019; date of current version March 27, 2020. This work was supported by the National Natural Science Foundation of China (61803259, 61833012). Recommended by Associate Editor C. Seatzu. *(Corresponding author: Xiang Yin.)*

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

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2019.2916940

proposed. For example, Darondeau *et al.* [14], [15], [30], [35], [38], [45] investigated how to synthesize a feedback supervisory controller to restrict the behavior the system such that the closed-loop system is opaque. In [19], [22], [39], [40], and [42], the problem of synthesizing insertion/edit functions that enforce opacity was studied. In [16], a runtime mechanism was used to enforce opacity K-step opacity. More recently, Barcelos and Basilio[20] proposed an approach for enforcing current-state opacity using events shuffle.

In this paper, we consider the problem of synthesizing *dynamic masks* that preserve *infinite-step opacity*. Dynamic mask is an information acquisition mechanism that acquires information from the system by dynamically turning ON/OFFthe associated sensors. Therefore, it can also be treated as a controller for sensors that can be controlled ON/OFFdynamically online. The dynamic mask synthesis problem is also referred to as the dynamic sensor activation problem in the DES literature; see, e.g., [9], [13], [34], [47], and [50]. The goal of this problem is to synthesize dynamic masks such that some system properties hold. In particular, in [8], the problem of synthesizing dynamic masks that preserve *current-state opacity* was investigated; an algorithm with an exponential complexity was also provided. The general idea is to reduce the synthesis problem to a two-player reach-avoid game in a suitably defined arena that captures all possible current-state information of the system.

As we discussed above, *infinite-step opacity* requires that the intruder should never know that the system was at a secret state for any specific previous instant with any possible future information. Therefore, to synthesize dynamic masks for infinite-step opacity, the following main difficulty arises. In the infinite-step opacity, whether or not a secret can be revealed not only depends on the current information, but also on the information in the future. This future information is known in the verification problem when the mask (either dynamic or static) is given. Therefore, the verification algorithms for infinite-step opacity proposed in [31] and [46] take the advantage that we can "borrow" the future information to check whether or not a secret can be revealed. However, in the synthesis problem, this future information depends on the sensing decisions in the future, which are unknown and to be synthesized. Moreover, the correctness of any future decision again depends on other actions in its further future. Therefore, this future dependency is, in fact, the key difference between the synthesis problem for the current-state opacity and the synthesis problem for infinite-step opacity.

In this paper, we tackle the problem of synthesizing dynamic masks for the infinite-step opacity by addressing the above-mentioned key difficulty. The main contributions of this paper are as follows. First, we propose a new type of information state in order to capture the delayed information in this synthesis problem. Specifically, each information state is used to represent the set of all possible delayed state estimates for all instants along the decision trajectory. We show that such a choice of information state summarizes all information needed in the synthesis problem in a finite domain. Moreover, the newly proposed information state can also be updated recursively upon the information change. Our new information state is more general than the subset-based information state that is widely used in partially observed synthesis problems related to current information. An effectively algorithm is then presented to solve the dynamic mask synthesis problem. Moreover, we show that the proposed novel information state is applicable to the synthesis problem for a class of properties with delayed information, including infinite-step K-anonymity and infinite-step indistinguishability, whose synthesis problems have never been considered in the literature.

To the best of our knowledge, most of the synthesis problems solved in the literature (for supervisory control, dynamic masks, or insertion functions) only consider the current-state-type property. One exception is the study in [30], where how to synthesize supervisors that enforce infinite-step opacity was investigated. The general idea of the study in [30] is to design a finite bank of supervisors where each supervisor enforces initial-state opacity for a type of current-state estimate encountered. Then, all supervisors in the bank work together to enforce infinite-step opacity, such that an event is disabled if some supervisor disables it. However, this "divide and conquer" approach does not work for the dynamic mask synthesis problem since dynamic masks do not restrict the behavior of the system; it does not make sense to obtain an overall dynamic mask from a bank of masks as the case of supervisors. Therefore, we need to handle all possible instant history and their dependencies within a single information structure using the approach proposed in the paper. Finally, applications of opacity can also be found in, for example, web services [4], location-based services [43], ship information systems [44], and mobile robots [28]. However, most of the applications only consider the current-state opacity. The proposed method can potentially be used to enforce the infinite-step opacity for these systems.

The rest of this paper is organized as follows. In Section II, we present some necessary preliminaries and formulate the dynamic mask synthesis problem for the infinite-step opacity. In Section III, we present a new class of information states for properties with delayed information. Using the newly proposed information state, we present a synthesis algorithm for the infinite-step opacity in Section IV. In Section V, we discuss how to reduce the complexity of the synthesis procedure and how to generalize the proposed approach to other properties. Finally, we conclude the paper in Section VI. Preliminary and partial versions of some of the results in this paper are presented in [48]. Compared with [48], this paper contains 1) all technical proofs omitted in [48]; 2) additional detailed explanations and examples; 3) an approach to further reduce the complexity of the synthesis algorithm; and (iv) discussion on how to generalize the proposed approach to other properties.

II. Preliminary

A. System Model

Let Σ be a finite set of events; a string $s = \sigma_1 \dots \sigma_n, \sigma_i \in \Sigma$ is a finite sequence of events, where $|s|$ denotes its length. We denote by Σ^* the set of all strings over Σ including the empty string ϵ. A language $L \subseteq \Sigma^*$ is a set of strings; $\overline{L} = \{s \in \Sigma^* : \exists t \in \Sigma^* \text{ s.t. } st \in L\}$ denotes its prefix-closure.

A DES is modeled as a finite-state automaton (FSA)

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

where X is the set of states, Σ is the set of events, $\delta : X \times \Sigma \to X$ is the transition function, and $X_0 \subseteq X$ is the set of initial states. Transition function δ is also extended to $X \times \Sigma^*$ in the usual manner by: $\forall x \in X, s \in \Sigma^*, \sigma \in \Sigma : \delta(x, s\sigma) = \delta(\delta(x, s), \sigma)$. For any state $x \in X$, we denote by $\mathcal{L}(G, x) = \{s \in \Sigma^* : \delta(x, s)!\}$ the set of all strings defined from x, where "!" means "is defined." We denote by $\mathcal{L}(G) = \cup_{x_0 \in X_0} \mathcal{L}(G, x_0)$ the language generated by G. We denote by $Acc(G)$ the *accessible part* of G; see, e.g., [7]. Let $G_1 = (X_1, \Sigma, \delta_1, X_{0,1})$ and $G_2 = (X_2, \Sigma, \delta_2, X_{0,2})$ be two FSAs. We say that G_1 is a *subautomaton* of G_2, denoted by $G_1 \sqsubseteq G_2$, if $X_1 \subseteq X_2, X_{0,1} \subseteq X_{0,2}$ and $\forall x_0 \in X_{0,1}, \forall s \in \mathcal{L}(G_1, x_0), \delta_2(x_0, s) = \delta_1(x_0, s)$.

In this paper, we consider a general *dynamic observation* mechanism. Specifically, we assume that the event set is partitioned as $\Sigma = \Sigma_o \dot\cup \Sigma_{uo} \dot\cup \Sigma_s$, where

1) Σ_o is the set of events whose occurrences can always be observed;
2) Σ_{uo} is the set of events whose occurrences never be observed; and
3) Σ_s is the set of events whose occurrences can *potentially* be observed. Whether or not the occurrence of an event in Σ_s can be observed depends on whether or not it is monitored, e.g., by turning ON its associated sensor.

A *sensing decision* θ, where $\Sigma_o \subseteq \theta \subseteq \Sigma_o \cup \Sigma_s$, is a set of events that determines which events are monitored. We denote by $\mathcal{O} := \{\theta \in 2^\Sigma : \Sigma_o \subseteq \theta \subseteq \Sigma_o \cup \Sigma_s\}$ the set of sensing decisions.

The observation of the system is then controlled by a *dynamic mask* $\Omega = (R, \Theta)$, where $R = (A_R, \Sigma, \delta_R, \{a_{0,R}\})$ is a deterministic automaton such that $\mathcal{L}(R) = \Sigma^*$ and $\Theta : A_R \to \mathcal{O}$ is a mapping that determines the sensing decision of each state. Furthermore, we require that

$$(\forall a, a' \in A_R, \sigma \in \Sigma : \delta_R(a, \sigma) = a')[a \neq a' \Rightarrow \sigma \in \Theta(a)].$$

This condition essentially says that the sensing decision can be updated (by updating the state of R) only when a monitored event occurs. In other words, events that are not monitored are defined as self-loops in R. Then, for any $s \in \mathcal{L}(G)$, $\Theta(\delta_R(a_{0,R}, s))$ is the set of currently observable (monitored) events; for the sake of simplicity, we also write $\Theta(\delta_R(a_{0,R}, s))$ by $\Omega(s)$.

The *projection* induced by a dynamic mask $\Omega = (R, \Theta)$ is a mapping $P_\Omega : \mathcal{L}(G) \to (\Sigma_o \cup \Sigma_s)^*$ defined recursively by: $\forall s \in \Sigma^*, \sigma \in \Sigma$

$$P_\Omega(\epsilon) = \epsilon, \quad P_\Omega(s\sigma) = \begin{cases} P_\Omega(s)\sigma & \text{if } \sigma \in \Omega(s) \\ P_\Omega(s) & \text{if } \sigma \notin \Omega(s) \end{cases}.$$

For any $L \subseteq \Sigma^*$, we define $P_\Omega(L) = \{P_\Omega(s) : s \in L\}$. Note that, for any $s \in \mathcal{L}(G)$, we have $\Omega(s) = \Omega(P_\Omega(s))$ since it is string $P_\Omega(s)$ that triggers state changes in Ω. Therefore, $\Omega(\alpha)$ also represents the current sensing decision upon the occurrence of any string s such that $P_\Omega(s) = \alpha$.

Let $\Omega = (R, \Theta)$ and $\Omega' = (R', \Theta')$ be two dynamic masks. We write that $\Omega' \leq \Omega$ if $\forall s \in \mathcal{L}(G) : \Omega'(s) \subseteq \Omega(s)$ and write that $\Omega' < \Omega$ if $\Omega' \leq \Omega$ and $\exists s \in \mathcal{L}(G) : \Omega'(s) \neq \Omega(s)$. Therefore, $\Omega' < \Omega$ implies that Ω acquires strictly more information than Ω'.

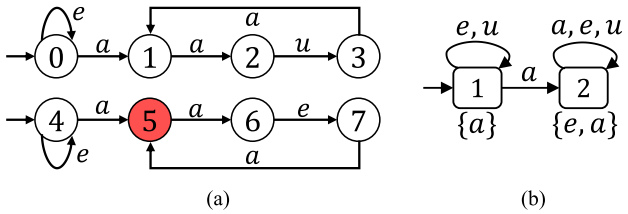

Fig. 1. For G, we have $\Sigma_o = \{a\}, \Sigma_{uo} = \{u\}, \Sigma_s = \{e\}$. The observation mapping $\Omega : A_R \to O$ is specified by each sensing decision associated with each state in R. (a) G with $X_S = \{5\}$. (b) $\Omega = (R, \Omega)$.

B. Infinite-Step Opacity

Let Ω be a dynamic mask and $\alpha\beta \in P_\Omega(\mathcal{L}(G))$ be an observable string. We define

$$\hat{X}_\Omega(\alpha \mid \alpha\beta) := \left\{ x \in X : \begin{array}{c} \exists x_0 \in X_0, \exists st \in \mathcal{L}(G, x_0) \\ \text{s.t. } x = \delta(x_0, s) \wedge \\ P_\Omega(s) = \alpha \wedge P_\Omega(\text{s.t.}) = \alpha\beta \end{array} \right\}.$$

Intuitively, $\hat{X}_\Omega(\alpha \mid \alpha\beta)$ is the *delayed state estimate* that captures the set of all possible states the system could be in at the instant when α is observed given the entire observation $\alpha\beta$, where β is a future information for the instant of α. We define $\hat{X}_\Omega(\alpha) := \hat{X}_\Omega(\alpha \mid \alpha)$ as the *current state estimate* upon the occurrence of $\alpha \in P_\Omega(\mathcal{L}(G))$.

The information acquired by the dynamic mask is usually transmitted to other modules for different purposes, e.g., for the purpose of supervisory control or for the purpose of fault diagnosis. However, the transmission channel between the dynamic mask and the utility module may be unreliable and could be "listened" by a passive intruder that is potentially malicious. In the context of opacity, we assume that the system has a "secret," which is modeled as a set of secret states $X_S \subset X$. We want that the secret should never be revealed to the intruder, which can access the information acquired by the dynamic mask. In particular, infinite-step opacity requires that the intruder should never know that the system was at a secret state for some specific instant, which is formally defined as follows.

Definition 1: System G is said to be *infinite-step opaque* w.r.t. X_S and Ω if $\forall \alpha\beta \in P_\Omega(\mathcal{L}(G)) : \hat{X}_\Omega(\alpha \mid \alpha\beta) \not\subseteq X_S$.

Example 1: Let us consider system G shown in Fig. 1(a) with $\Sigma_o = \{a\}, \Sigma_{uo} = \{u\}$, and $\Sigma_s = \{e\}$. Let us consider dynamic mask Ω shown in Fig. 1(b). Assume that state 5 is the unique secret state, i.e., $X_S = \{5\}$. Then, G is not infinite-step opaque w.r.t. Ω and X_S. To see this, we consider string $eaae \in \mathcal{L}(G)$ with $P_\Omega(eaae) = aae$. Note that the first occurrence of event e is not observable since the sensing decision at state 1 in R is $\{a\}$, while the second occurrence of e is observable since the dynamic mask is then at state 2. Then, we know that $\hat{X}_\Omega(a \mid aae) = \{5\} \subseteq X_S$, i.e., the system is not infinite-step opaque. Intuitively, this says that when string aae is observed, we know for sure that the system was at a secret state two steps ago.

Given a system G, algorithms have been proposed in [31] and [46] for the *verification* of infinite-step opaque for the case of static observation; the algorithms can be easily modified for the case of dynamic observation when Ω is given. In this paper, we consider the *synthesis* problem, i.e., we want to *synthesize* a dynamic mask that controls the information acquisition process such that the entire system is infinite-step opaque. Clearly, the less information acquired, the system is more likely to be opaque. An extreme case is to consider a dynamic mask that never monitors any event; such a system is clearly infinite-step opaque. However, in addition to the security constraint, dynamic mask is also an interface that acquires information for the user. Therefore, for the purpose of *utility*, we also want that the dynamic mask monitors as many events as possible when the opacity constraint is satisfied. To this end, we formulate the optimal dynamic mask synthesis problem that we solve in this paper.

Problem 1: Given system G and a set of secret states $X_S \subset X$, synthesize a dynamic mask Ω such that

1) G is infinite-step opaque w.r.t. X_S and Ω; and
2) For any Ω' satisfying 1), we have $\Omega \not< \Omega'$.

Remark 1: Note that, for the synthesized solution, we require that $\forall \Omega' : \Omega \not< \Omega'$, which is weaker than $\forall \Omega' : \Omega' < \Omega$. In the literature, the former is referred to as a *maximal* solution, while the latter is referred to as the *supremal* solution. However, it is well-known [33] that the supremal solution does not exist in general in the sensor activation problem. Instead, it is possible that there are two incomparable maximal solutions Ω_1 and Ω_2, i.e., none of them can be improved but $\Omega_i \not< \Omega_j, i \neq j$. Our goal here is to find *one* maximal solution for infinite-step opacity. There may also have other maximal solutions that are *incomparable* with our solution, but none of them can be strictly better than the synthesized one.

Finally, we introduce several operators that will be used in the paper. Let $q \in 2^X$ be a set of states, $\theta \in \mathcal{O}$ be a sensing decision, and $\sigma \in \Sigma_o \cup \Sigma_s$ be an event. We define

$$UR_\theta(q) = \{x \in X : \exists x' \in q, \exists w \in (\Sigma \setminus \theta)^* \text{ s.t. } x = \delta(x', w)\}$$
$$NX_\sigma(q) = \{x \in X : \exists x' \in q \text{ s.t. } x = \delta(x', \sigma)\}.$$

Intuitively, $UR_\theta(q)$ is the set of states that can be reached from states in q via strings that are unobservable under θ; and $NX_\sigma(q)$ is the set of states that can be reached *immediately* from states in q upon the occurrence of σ.

Example 2: Let us still consider system G shown in Fig. 1(a). Let $\{2, 6\} \in 2^X$ be a set of states. Then, upon sensing decision $\{a\} \in \mathcal{O}$, we have $UR_{\{a\}}(\{2, 6\}) = \{2, 3, 6, 7\}$. Then, upon observable event $a \in \Sigma_o \cup \Sigma_s$, we have $NX_a(\{2, 3, 6, 7\}) = \{1, 5\}$.

Similarly, let $\rho \in 2^{X \times X}$ be a set of state pairs, $\theta \in \mathcal{O}$ be a sensing decision, and $\sigma \in \Sigma_o \cup \Sigma_s$ be an event. We define

$$\widetilde{UR}_\theta(\rho) = \left\{(x_1, x_3) \in X \times X : \begin{array}{c}\exists (x_1, x_2) \in \rho, w \in (\Sigma \setminus \theta)^* \\ \text{s.t. } x_3 = \delta(x_2, w)\end{array}\right\}$$
$$\widetilde{NX}_\sigma(\rho) = \{(x_1, x_3) \in X \times X : \exists (x_1, x_2) \in \rho, x_3 = \delta(x_2, \sigma)\}.$$

Note that, each element in ρ is a *state pair*; we use the second state in the pair to represent the current state of the system and use the first state in the pair to represent where the current state comes from. This is why, in the definitions of $\widetilde{UR}_\theta(\rho)$ and $\widetilde{NX}_\sigma(\rho)$, only states in the second components are changed.

Still, let $q \in 2^X$ be a set of states. We define

$$\odot_\theta(q) = \{(x, x') \in q \times q : \exists w \in (\Sigma \setminus \theta)^* \text{ s.t. } \delta(x, w) = x'\}.$$

Intuitively, $\odot_\theta(q)$ maps q to a set of state pairs such that, in each pair of states, the first state can reach the second state via strings that are unobservable under θ.

Example 3: We still consider system G shown in Fig. 1(a). Let $\{(1, 2), (5, 6)\} \in 2^{X \times X}$ be a set of state pairs, which represents that the system is either 1) currently at state 2 by initialing from state 1; or 2) currently at state 6 by initialing from state 5. Then, upon sensing decision $\{a\} \in \mathcal{O}$, we have $UR_{\{a\}}(\{(1, 2), (5, 6)\}) = \{(1, 2), (1, 3), (5, 6), (5, 7)\}$. Then, upon observable event $a \in \Sigma_o \cup \Sigma_s$, we have $NX_a(\{(1, 2), (1, 3), (5, 6), (5, 7)\}) = \{(1, 1), (5, 5)\}$. Additionally, for $\{2, 3, 6, 7\} \in 2^X$, $\{a\} \in \mathcal{O}$, we have $\odot_{\{a\}}(\{2, 3, 6, 7\}) = \{(2, 2), (2, 3), (3, 3), (6, 6), (6, 7), (7, 7)\}$.

III. Information State for Infinite-Step Opacity

In this section, we discuss how to select suitable information state for infinite-step opacity and describe the information evolution of a dynamic mask.

A. Choice of Information State

Given a dynamic mask $\Omega = (R, \Theta)$, it works as follows. Initially, it provides an initial sensing decision $\Omega(\epsilon)$. Then, only some feasible and monitored event $\sigma \in \Omega(\epsilon)$ can be observed. Upon the occurrence of σ, the transition function of R is triggered and it changes its sensing decision to $\Omega(\sigma)$ and waits for the occurrence of the next monitored event, and so forth.

Let $\alpha = \sigma_1 \sigma_2 \ldots \sigma_n \in P_\Omega(\mathcal{L}(G))$ be an observable string. Then, the information available upon the occurrence of α is an alternating sequence, called a *run*, defined by

$$\mathcal{R}_\Omega(\alpha) := \Omega(\epsilon)\sigma_1\Omega(\sigma_1)\sigma_2 \ldots \sigma_n\Omega(\sigma_1 \ldots \sigma_n). \quad (1)$$

Although a run contains the complete information available, it requires infinite memory when the length of the observed string goes to infinite. In order to efficiently solve the synthesis problem, we need more compact ways to summarize the information available; this is usually referred to as *information state* in the systems theory. Roughly speaking, information states are (mostly finite) sufficient statistics that contain all useful (potentially infinite) information. The choice of information states is problem dependent. For example, in many synthesis problems for a partially observed DES, 2^X is a suitable set of information states when only current-state-type properties are considered; see, e.g., the supervisory control problem [45] and the sensor activation problems for current-state opacity [8] and diagnosability [13]. However, 2^X is not sufficient for our purpose, since infinite-step opacity not only requires that the secret is not revealed currently, but also requires that the secret will not be revealed at any instant in the future.

In this paper, we propose to use the following set of information states:

$$I := 2^X \times 2^{2^{X\times X}} \times \mathcal{O}$$

and each information state $\imath \in I$ has the form of $\imath = (C(\imath), D(\imath), O(\imath))$ such that

1) the first component $C(\imath) \in 2^X$ is a set of states representing the *current state estimate* of the system.
2) the second component $D(\imath) \in 2^{2^{X\times X}}$ is a set of state-pair-sets representing all possible *delayed state estimates* for all previous instants. More specifically, each element $\rho \in D(\imath)$ is a set of state pairs in the form of $\rho = \{(x_1, x_1'), \ldots, (x_{|\rho|}, x_{|\rho|}')\} \in 2^{X\times X}$, where x_i represents a state the system could be in at some previous instant and x_i' represents a state the system could be in currently from x_i; and set $D(\imath)$ contains all possible such ρ for all previous instants.
3) the third component $O(\imath) \in \mathcal{O}$ is a sensing decision representing the set of events being monitored currently.

B. Information State Evolution

Next, we describe how information states evolve. Now, suppose that we are at information-state

$$\imath = (C(\imath), D(\imath), O(\imath)) \in 2^X \times 2^{2^{X\times X}} \times \mathcal{O}$$

and suppose that a monitored event $\sigma \in O(\imath)$ is observed and a new sensing decision $\theta \in \mathcal{O}$ is made immediately after the occurrence of σ. Then, we move to a new information state $\imath' = (C(\imath'), D(\imath'), O(\imath'))$ as follows:

$$\begin{cases} C(\imath') = & UR_\theta(NX_\sigma(C(\imath))) \\ D(\imath') = & \{\widetilde{UR}_\theta(\widetilde{NX}_\sigma(\rho)) \in 2^{X\times X} : \rho \in D(\imath)\} \\ & \cup\{\odot_\theta(C(\imath'))\} \\ O(\imath') = & \theta \end{cases}. \quad (2)$$

Equation (2) is the key of this paper and it is also referred to as the *information state updating rule* and we denote by $\imath \xrightarrow{(\sigma,\theta)} \imath'$ if $\imath, \imath'$ and (σ, θ) satisfy this rule. Intuitively, $C(\imath')$ is the updated current state estimate upon the observation of σ and the newly issue sensing decision θ. This is done by the unobservable reach as the standard observer automaton. The computation of $D(\imath')$ is more involved. Specifically, the first part essentially *smooths* the delayed state estimates for all previous instants using the new information obtained and the second part mainly adds the current state estimate, which will become delayed state estimate for future instants. Finally, $O(\imath')$ simply remembers the latest sensing decision θ.

Now, let Ω be a dynamic mask and $\alpha = \sigma_1 \ldots \sigma_n \in P_\Omega(\mathcal{L}(G))$ be an observable string. Then, run $\mathcal{R}_\Omega(\alpha)$ induces the following information states evolution:

$$\imath_0 \xrightarrow{(\sigma_1,\Omega(\sigma_1))} \imath_1 \xrightarrow{(\sigma_2,\Omega(\sigma_1\sigma_2))} \cdots \xrightarrow{(\sigma_n,\Omega(\sigma_1\ldots\sigma_n))} \imath_n$$

where

$$\imath_0 = (UR_{\Omega(\epsilon)}(X_0), \{\odot_{\Omega(\epsilon)}(UR_{\Omega(\epsilon)}(X_0))\}, \Omega(\epsilon)) \quad (3)$$

is the initial information state. We denote by $\mathcal{I}_\Omega(\alpha)$ the information state reached by the run of α, i.e., $\mathcal{I}_\Omega(\alpha) = \imath_n$.

Example 4: Again, let us consider system G in Fig. 1(a) and dynamic mask Ω in Fig. 1(b). Let us consider observable string $aa \in P_\Omega(\mathcal{L}(G))$. Initially, we have

$$\begin{aligned}\mathcal{I}_\Omega(\epsilon) &= (UR_{\Omega(\epsilon)}(X_0), \{\odot_{\Omega(\epsilon)}(UR_{\Omega(\epsilon)}(X_0))\}, \Omega(\epsilon)) \\ &= (\{0,4\}, \{\{(0,0),(4,4)\}\}, \{a\}).\end{aligned}$$

Once event a is observed and new sensing decision $\Omega(a) = \{e, a\}$ is made, the information state is updated to

$$\mathcal{I}_\Omega(a) = (\{1,5\}, \{\{(0,1),(4,5)\}, \{(1,1),(5,5)\}\}, \{e,a\})$$

which is computed by

$$\begin{aligned} C(\mathcal{I}_\Omega(a)) &= UR_{\{e,a\}}(NX_a(\{0,4\})) = \{1,5\} \\ D(\mathcal{I}_\Omega(a)) &= \{\widetilde{UR}_{\{e,a\}}(\widetilde{NX}_a(\{(0,0),(4,4)\}))\} \\ &\quad \cup\{\odot_{\{e,a\}}(\{1,5\})\} \\ &= \{\{(0,1),(4,5)\}\} \cup \{\{(1,1),(5,5)\}\} \\ O(\mathcal{I}_\Omega(a)) &= \{e,a\}. \end{aligned}$$

Similarly, we have $\mathcal{I}_\Omega(aa) = (\{2,3,6\}, \{\{(0,2),(0,3),(4,6)\}, \{(1,2),(1,3),(5,6)\}, \{(2,2),(2,3),(3,3),(6,6)\}\}, \{e,a\})$, which is computed by

$$\begin{aligned} C(\mathcal{I}_\Omega(aa)) &= UR_{\{e,a\}}(NX_a(\{1,5\})) = \{2,3,6\} \\ D(\mathcal{I}_\Omega(aa)) &= \{\widetilde{UR}_{\{e,a\}}(\widetilde{NX}_a(\{(0,1),(4,5)\}))\} \\ &\quad \cup\{\widetilde{UR}_{\{e,a\}}(\widetilde{NX}_a(\{(1,1),(5,5)\}))\} \\ &\quad \cup\{\odot_{\{e,a\}}(\{2,3,6\})\} \\ &= \{\{(0,2),(0,3),(4,6)\}\} \cup \{\{(1,2),(1,3),(5,6)\}\} \\ &\quad \cup\{\{(2,2),(2,3),(3,3),(6,6)\}\} \\ O(\mathcal{I}_\Omega(aa)) &= \{e,a\}. \end{aligned}$$

Next, we show that the information states selected together with the updating rule defined in (2) and the initial information state defined in (3) indeed capture all relevant information in the infinite-step opacity synthesis problem. First, we characterize each component in $\mathcal{I}_\Omega(\alpha)$.

Proposition 1: Let Ω be a dynamic mask, $\alpha \in P_\Omega(\mathcal{L}(G))$ be an observable string, and $\mathcal{I}_\Omega(\alpha)$ be the information state reached. Then, we have

1) $C(\mathcal{I}_\Omega(\alpha)) = \hat{X}_\Omega(\alpha)$; and
2) $D(\mathcal{I}_\Omega(\alpha)) = \{\rho_{\beta,\alpha} \in 2^{X\times X} : \beta \in \overline{\{\alpha\}}\}$, where

$$\rho_{\beta,\alpha} = \left\{ (x,x') \in X\times X : \begin{array}{r} \exists x_0 \in X_0, st \in \mathcal{L}(G,x_0) \text{ s.t.} \\ P_\Omega(s) = \beta \wedge P_\Omega(st) = \alpha \wedge \\ \delta(x_0,s) = x \wedge \delta(x_0,st) = x' \end{array} \right\}.$$

Proof: In the first component of the updating rule, $UR_\theta(NX_\sigma(C(\cdot)))$ is actually the recursive computation of the current state estimate in the dynamic observation setting; see, e.g., [13]. This gives 1). Hereafter, we prove 2) by induction on the length of α.

Induction Basis: Suppose that $|\alpha| = 0$, i.e., $\alpha = \epsilon$. Then, we know that $\mathcal{I}_\Omega(\epsilon) = \imath_0$ and $D(\mathcal{I}_\Omega(\epsilon)) =$

$\{\odot_{\Omega(\epsilon)}(UR_{\Omega(\epsilon)}(X_0))\}$, where

$$\begin{aligned}
&\odot_{\Omega(\epsilon)}\left(UR_{\Omega(\epsilon)}(X_0)\right)\\
&=\left\{(x,x')\in X\times X:\begin{array}{c}\exists x\in UR_{\Omega(\epsilon)}(X_0), w\in(\Sigma\setminus\Omega(\epsilon))^*\\ \text{s.t. } \delta(x,w)=x'\end{array}\right\}\\
&=\left\{(x,x')\in X\times X:\begin{array}{c}\exists x_0\in X_0, \exists st\in(\Sigma\setminus\Omega(\epsilon))^*\\ \text{s.t. } \delta(x_0,s)=x\wedge\delta(x,t)=x'\end{array}\right\}\\
&=\left\{(x,x')\in X\times X:\begin{array}{c}\exists x_0\in X_0, \exists st\in\mathcal{L}(G,x_0)\text{ s.t.}\\ P_\Omega(s)=P_\Omega(st)=\epsilon\wedge\\ \delta(x_0,s)=x\wedge\delta(x_0,st)=x'\end{array}\right\}\\
&=\rho_{\epsilon,\epsilon}.
\end{aligned}$$

Therefore, $D(\mathcal{I}_\Omega(\epsilon))=\{\rho_{\epsilon,\epsilon}\}$ and the induction basis holds.

Induction Step: Now, let us assume that, for $|\alpha|=k$, 2) holds. We need to prove that 2) still holds for any $\alpha\sigma\in P_\Omega(\mathcal{L}(G))$, where $|\alpha|=k$ and $\sigma\in\Sigma_o\cup\Sigma_s$.

By the updating rule in (2), we know that

$$\begin{aligned}
D(\mathcal{I}_\Omega(\alpha\sigma))=\{\widetilde{UR}_{\Omega(\alpha\sigma)}(\widetilde{NX}_\sigma(\rho))\in 2^{X\times X}:\rho\in D(\mathcal{I}_\Omega(\alpha))\}\\
\cup\{\odot_{\Omega(\alpha\sigma)}(\hat{X}_\Omega(\alpha\sigma))\}.
\end{aligned}\tag{4}$$

Since $|\alpha|=k$, by the induction hypothesis, we know

$$D(\mathcal{I}_\Omega(\alpha))=\{\rho_{\beta,\alpha}\in 2^{X\times X}:\beta\in\overline{\{\alpha\}}\}.\tag{5}$$

Additionally, by the definition of $\widetilde{UR}$ and $\widetilde{NX}$, we have

$$\begin{aligned}
&\widetilde{UR}_{\Omega(\alpha\sigma)}(\widetilde{NX}_\sigma(\rho_{\beta,\alpha}))\\
&=\left\{(x_1,x_3)\in X\times X:\begin{array}{c}\exists(x_1,x_2)\in\rho_{\beta,\alpha}, \exists w\in(\Sigma\setminus\Omega(\alpha\sigma))^*\\ \text{s.t. } x_3=\delta(x_2,\sigma w)\end{array}\right\}\\
&=\left\{(x_1,x_3)\in X\times X:\begin{array}{c}\exists x_0\in X_0, \exists st\in\mathcal{L}(G,x_0),\\ \exists w\in(\Sigma\setminus\Omega(\alpha\sigma))^*\text{ s.t.}\\ P_\Omega(s)=\beta\wedge P_\Omega(st)=\alpha\wedge\\ \delta(x_0,s)=x_1\wedge\delta(x_0,st\sigma w)=x_3\end{array}\right\}\\
&=\left\{(x_1,x_3)\in X\times X:\begin{array}{c}\exists x_0\in X_0, \exists st'\in\mathcal{L}(G,x_0)\text{ s.t.}\\ P_\Omega(s)=\beta\wedge P_\Omega(st')=\alpha\sigma\\ \wedge\delta(x_0,s)=x_1\wedge\delta(x_0,st')=x_3\end{array}\right\}\\
&=\rho_{\beta,\alpha\sigma}.
\end{aligned}\tag{6}$$

Combining (5) and (6), we have

$$\begin{aligned}
&\{\widetilde{UR}_{\Omega(\alpha\sigma)}(\widetilde{NX}_\sigma(\rho))\in 2^{X\times X}:\rho\in D(\mathcal{I}_\Omega(\alpha))\}\\
&=\{\widetilde{UR}_{\Omega(\alpha\sigma)}(\widetilde{NX}_\sigma(\rho_{\beta,\alpha}))\in 2^{X\times X}:\beta\in\overline{\{\alpha\}}\}\\
&=\{\rho_{\beta,\alpha\sigma}\in 2^{X\times X}:\beta\in\overline{\{\alpha\}}\}.
\end{aligned}\tag{7}$$

Moreover, we have

$$\begin{aligned}
&\odot_{\Omega(\alpha\sigma)}\left(\hat{X}_\Omega(\alpha\sigma)\right)\\
&=\left\{(x,x')\in X\times X:\begin{array}{c}\exists x\in\hat{X}_G(\alpha\sigma), \exists w\in(\Sigma\setminus\Omega(\alpha\sigma))^*\\ \text{s.t. } \delta(x,w)=x'\end{array}\right\}\\
&=\left\{(x,x')\in X\times X\exists w\in(\Sigma\setminus\Omega(\alpha\sigma))^*\text{ s.t. }\begin{array}{c}\exists x_0\in X_0, \exists s\in\mathcal{L}(G,x_0),\\ P_\Omega(s)=\alpha\sigma\\ \wedge\delta(x_0,s)=x\wedge\delta(x_0,sw)=x'\end{array}\right\}\\
&=\left\{(x,x')\in X\times X:\begin{array}{c}\exists x_0\in X_0, \exists sw\in\mathcal{L}(G,x_0)\text{ s.t.}\\ P_\Omega(s)=P_\Omega(sw)=\alpha\sigma\\ \wedge\delta(x_0,s)=x\wedge\delta(x_0,sw)=x'\end{array}\right\}\\
&=\rho_{\alpha\sigma,\alpha\sigma}.
\end{aligned}\tag{8}$$

Therefore, by combining (4), (7), and (8), we have

$$\begin{aligned}
D(\mathcal{I}_\Omega(\alpha\sigma))&=\{\rho_{\beta,\alpha\sigma}\in 2^{X\times X}:\beta\in\overline{\{\alpha\}}\}\cup\{\rho_{\alpha\sigma,\alpha\sigma}\}\\
&=\{\rho_{\beta,\alpha\sigma}\in 2^{X\times X}:\beta\in\overline{\{\alpha\sigma\}}\}.
\end{aligned}$$

This completes the induction step. ■

Recall that, each information state $\imath$ is in the form of $\imath=(C(\imath),D(\imath),O(\imath))$, where $D(\imath)$ is a set of state-pair-sets. Then, we define

$$D_1(\imath):=\{\{x\in X:(x,x')\in\rho\}:\rho\in D(\imath)\}$$

which consists of the first component of each state pairs in $D(\imath)$. For example, for $\imath=(\{1,5\},\{\{(0,1),(4,5)\},\{(1,1),(5,5)\}\},\{e,a\})$, we have $D_1(\imath)=\{\{0,4\},\{1,5\}\}$. Then, we have the following corollary.

Corollary 1: Let Ω be a dynamic mask, $\alpha\in P_\Omega(\mathcal{L}(G))$ be an observable string, and $\mathcal{I}_\Omega(\alpha)$ be the information state reached. Then, we have

$$D_1(\mathcal{I}_\Omega(\alpha))=\{\hat{X}_G(\beta\mid\alpha)\in 2^X:\beta\in\overline{\{\alpha\}}\}.$$

Proof: By Proposition 1, we know that $D(\mathcal{I}_\Omega(\alpha))=\{\rho_{\beta,\alpha}\in 2^{X\times X}:\beta\in\overline{\{\alpha\}}\}$. Therefore

$$\begin{aligned}
&D_1(\mathcal{I}_\Omega(\alpha))\\
&=\left\{\{x\in X:(x,x')\in\rho_{\beta,\alpha}\}:\beta\in\overline{\{\alpha\}}\right\}\\
&=\left\{\left\{x\in X:\begin{array}{c}\exists x_0\in X_0, \exists st\in\mathcal{L}(G,x_0)\\ \text{s.t. } x=\delta(x_0,s)\\ \wedge P_\Omega(s)=\beta\wedge P_\Omega(st)=\alpha\end{array}\right\}:\beta\in\overline{\{\alpha\}}\right\}\\
&=\left\{\hat{X}_G(\beta\mid\alpha)\in 2^X:\beta\in\overline{\{\alpha\}}\right\}.
\end{aligned}\tag{9}$$

This completes the proof. ■

Corollary 1 tells that, using the proposed information state updating rule, $D_1(\mathcal{I}_\Omega(\alpha))$ can capture all possible delayed state estimates for all possible previous instants. Let us define

$$I_{\text{bad}}=\{\imath\in I:\exists q\in D_1(\imath)\text{ s.t. }q\subseteq X_S\}\tag{10}$$

as the set of information states in which infinite-step opacity is violated for some previous instant. Then, to check whether or

not a given dynamic mask Ω is infinite-step opaque, it suffices to check whether or not Ω can reach some information state in $I_{\rm bad}$.

Finally, we would like to emphasize that the information-state-based characterization is not the most efficient way for the purpose of *verification* of infinite-step opacity. In particular, to verify infinite-step opacity, we can check, for each possible current state estimate, whether or not any string that comes from a secret state in it has a string that comes from a nonsecret state in the current state estimate such that they have the same projection; see, e.g., [31] and [46]. This leads to an exponential complexity for the verification of infinite-step opacity rather than the doubly exponential space of our information states. However, the idea of the above verification procedure *cannot* be applied to the synthesize problem, since the existence of such an observational equivalent "nonsecret" depends on the sensing decisions in the future, which are unknown and are to be determined in the synthesis problem. Therefore, we cannot "borrow" this future information as one can do in the verification problem. This is, in fact, the main difficulty in the infinite-opacity synthesis problem, which is addressed by the new type of information state proposed in this paper.

IV. Synthesis Procedure of Infinite-Step Opacity

In this section, we first present an approach to synthesize a dynamic mask that preserves infinite-step opacity and then prove its correctness.

A. Synthesis Algorithm

As we discussed earlier, to synthesize a dynamic mask Ω satisfying the infinite-step opacity requirement, it suffices to guarantee, by construction, that any run induced will not reach a bad information state in $I_{\rm bad}$.

To this end, first, we define an FSA

$$\mathbb{T} = (A_{\mathbb{T}}, \Sigma_{\mathbb{T}}, \delta_{\mathbb{T}}, A_{0,\mathbb{T}})$$

that captures all possible runs and all possible reachable information states. Specifically

1) $A_{\mathbb{T}} = I = 2^X \times 2^{2^{X\times X}} \times \mathcal{O}$ is the set of states;
2) $\Sigma_{\mathbb{T}} = (\Sigma_o \cup \Sigma_s) \times \mathcal{O}$ is the set of events;
3) $\delta_{\mathbb{T}} : A_{\mathbb{T}} \times \Sigma_{\mathbb{T}} \to A_{\mathbb{T}}$ is the transition function defined by: for any $\imath, \imath' \in A_{\mathbb{T}}, (\sigma,\theta) \in \Sigma_{\mathbb{T}}$, $\delta_{\mathbb{T}}(\imath,(\sigma,\theta)) = \imath'$ if $\sigma \in O(\imath)$ and $\imath \xrightarrow{(\sigma,\theta)} \imath'$;
4) $A_{0,\mathbb{T}}$ is the set of initial states defined by

$$A_{0,\mathbb{T}} = \{(UR_\theta(X_0), \{\odot_\theta(UR_\theta(X_0))\}, \theta) \in A_{\mathbb{T}} : \theta \in \mathcal{O}\}.$$

Intuitively, $\mathbb{T}$ captures all possible information state evolutions since its transition function is defined for any information states that satisfy the updating rule. The initial states of $\mathbb{T}$ are not unique in order to capture all possible initial sensing decisions. Automaton $\mathbb{T}$ will serve as the basis for synthesizing a dynamic mask satisfying infinite-step opacity. This structure will not be constructed explicitly. Instead, we will construct a subautomaton of $\mathbb{T}$ directly as a solution to Problem 1. First, we introduce some necessary notations and concepts.

Let $T \sqsubseteq \mathbb{T}$ be a subautomaton of $\mathbb{T}$, where $T = (A_T, \Sigma_T, \delta_T, A_{0,T})$. For each state $\imath \in A_T$, we define

$$\Sigma_T^{\rm succ}(\imath) := \{\sigma \in O(\imath) : \exists \theta \in \mathcal{O} \text{ s.t. } \delta_T(\imath,(\sigma,\theta))!\}$$

as the set of observable events defined at $\imath$. For each observable event $\sigma \in \Sigma_T^{\rm succ}(\imath)$, we define

$$\Theta_T(\sigma,\imath) := \{\theta \in \mathcal{O} : \delta_T(\imath,(\sigma,\theta))!\}$$

as the set of sensing decisions associated with σ at state $\imath$. Note that, since the transition function of $\mathbb{T}$ is defined for all events satisfying the updating rule, $\Sigma_{\mathbb{T}}^{\rm succ}(\imath)$ is the set of all possible events that can be observed from $\imath$. We say that subautomaton $T \sqsubseteq \mathbb{T}$ is

1) *safe*, if $A_T \cap I_{\rm bad} = \emptyset$;
2) *observation consistent*, if $\forall \imath \in A_T, \Sigma_T^{\rm succ}(\imath) = \Sigma_{\mathbb{T}}^{\rm succ}(\imath)$;
3) *decision deterministic*, if $|A_{0,T}| = 1$ and $\forall \imath \in A_T, \forall \sigma \in \Sigma_T^{\rm succ}(\imath) : |\Theta_T(\sigma,\imath)| = 1$.

Intuitively, observation consistency says that any feasible observation should be defined (associated with some θ) at each state in T. This captures the fact that a dynamic mask should be able to react to any possible observation. Decision determinism essentially says that 1) the initial sensing decision is unique; and 2) upon the occurrence of a new observable event σ, the next sensing decision is also unique, which is the unique sensing decision associated with σ. Therefore, any safe, observation consistent and decision deterministic subautomaton $T \sqsubseteq \mathbb{T}$ can be modified as a dynamic mask that satisfies the infinite-step opacity requirement.

On the basis of the above discussion, Algorithm 1 is proposed to solve Problem 1. Let us explain how Algorithm 1 works. Lines 1–4 aim to explore the entire reachable space of $I \setminus I_{\rm bad}$ represented by automaton T, whicis a subautomaton of $\mathbb{T}$. Specifically, line 1 defines the initial configuration of T with initial states representing all possible initial sensing decisions that do not violate infinite-step opacity initially. Then, procedure EXPAND in lines 24–34 simply traverses the entire reachable space of $I \setminus I_{\rm bad}$ by a depth-first-search implemented recursively. Note that the resulting automaton T after line 4 may not be observation consistent since some events σ may lead to states in $I_{\rm bad}$ no matter what θ they are associated with. Therefore, the while-loop in lines 5–7 mainly removes states that violate observation consistency from A_T. Note that removing one state from T may destroy the observation consistency of other states; hence, the while-loop may execute at most $|A_T|$ times and the resulting T is the largest safe and observation consistent subautomaton of $\mathbb{T}$. In the worst case, all initial states in T can be removed. This implies that opacity cannot be satisfied by any sensing strategy; this is implemented by lines 8–10.

Now, suppose that automaton T obtained after line 7 contains initial states. However, T may still not be decision deterministic, i.e., for some $\imath \in A_T, \sigma \in \Sigma_T^{\rm succ}(\imath)$, σ may be associated with multiple sensing decisions. Therefore, lines 11–18 mainly aim to find a decision deterministic subautomaton of T. Specifically, lines 11 and 12 select a single initial state from $A_{0,T}$ and, similarly, lines 13–18 select a single sensing decision for each observable event at each state in A_T. Moreover, in lines 11 and 15, we select sensing decisions that are *local maximal*; this

Algorithm 1: Synthesize Dynamic Mask $\Omega = (T, \Theta)$.

1: Define a FSA $T = (A_T, \Sigma_T, \delta_T, A_{0,T})$ with $A_T = A_{0,T} = \{(UR_\theta(X_0), \{\odot_\theta(UR_\theta(X_0))\}, \theta) \in X_T : \theta \in \mathcal{O}\} \setminus I_{\rm bad}$ and δ_T is undefined for any transition
2: **for** $\imath_0 \in A_{0,T}$ **do**
3: EXPAND($\imath_0, T$)
4: **end for**
5: **while** $\exists \imath \in A_T : \Sigma_T^{\rm succ}(\imath) \neq \Sigma_{\mathbb{T}}^{\rm succ}(\imath)$ **do**
6: Remove $\imath$ from A_T and remove its associated transitions from δ_T
7: **end while**
8: **if** $A_{0,T} = \emptyset$ **then**
9: **return** "No Solution"
10: **end if**
11: Find $\imath_0 \in A_{0,T}$ such that $\forall \imath_0' \in A_{0,T} : O(\imath_0) \not\subset O(\imath_0')$
12: Re-define initial states of T by $A_{0,T} \leftarrow \{\imath_0\}$
13: **for** $\imath \in A_T$ **do**
14: **for** $\sigma \in \Sigma_T^{\rm succ}(\imath)$ **do**
15: Find $\theta \in \Theta_T(\sigma, \imath)$ s.t. $\forall \theta' \in \Theta_T(\sigma, \imath) : \theta \not\subset \theta'$
16: At state $\imath$, remove all transitions labeled with $(\sigma, \theta'), \theta' \neq \theta$
17: **end for**
18: **end for**
19: $T \leftarrow Acc(T)$
20: Redefine $\Sigma_T = \Sigma$ and rename each event (σ, θ) by σ
21: Make the transition function of T total by adding self-loops for undefined events
22: Define mapping $\Theta : A_T \to \mathcal{O}$ by $\forall \imath \in A_T : \Theta(\imath) = O(\imath)$
23: **return** $\Omega = (T, \Theta)$
24: **procedure** EXPAND($\imath, T$)
25: **for** $(\sigma, \theta) \in (\Sigma_o \cup \Sigma_s) \times \mathcal{O} : \delta_{\mathbb{T}}(\imath, (\sigma, \theta)) = \imath'$ **do**
26: **if** $\imath' \notin I_{\rm bad}$ **then**
27: add transition $\delta_T(\imath, (\sigma, \theta)) = \imath'$ to T
28: **if** $\imath' \notin A_T$ **then**
29: $A_T \leftarrow A_T \cup \{\imath'\}$
30: EXPAND($\imath', T$)
31: **end if**
32: **end if**
33: **end for**
34: **end procedure**

guarantees, by construction, that no sensing decision can improve the selected one. By removing transitions from T, some states may not be reachable anymore and line 19 takes the accessible part of T. After line 19, we have obtained a safe, observation consistent and decision deterministic subautomaton of $\mathbb{T}$, i.e., T. Then, lines 20–22 modify it as a dynamic mask. Specifically, line 20 erases the sensing decision associated with each observation (this decision has already been encoded in the last component of each state in T). Line 21 simply adds self-loops for either infeasible events or unmonitored events so that $\mathcal{L}(T) = \Sigma^*$. Finally, we define the output mapping Θ by the last component of each state in T and return $\Omega := (T, \Theta)$ as the desired dynamic mask.

We will formally show later the properties of Algorithm 1 and prove its correctness. First, we illustrate Algorithm 1 by the following example.

Example 5: Again, let us consider system G in Fig. 1(a) with $\Sigma_o = \{a\}, \Sigma_{uo} = \{u\}, \Sigma_s = \{e\}$, and $X_S = \{5\}$. We want to synthesize a dynamic mask Ω satisfying infinite-step opacity. Note that $\mathcal{O} = \{\{a\}, \{e, a\}\}$, i.e., there are two choices for sensing decision, monitor event e or not. We apply Algorithm 1 to this system. Specifically, after the execution of line 4, we obtain T shown in the box marked with blue-dashed lines in Fig. 2. For the sake of clarity, we denote this intermediate automaton by T_1 and we rename each state in T_1 by $S_1, \ldots, S_{10}$ as shown in the figure. At state S_7, if we take transition $(a, \{e, a\})$, then state $\imath' = (\{7\}, \{\{(4, 7)\}, \{(5, 7)\}, \{(6, 7)\}, \{(7, 7)\}\}, \{e, a\})$ is reached. However, since $D_1(\imath') = \{\{4\}, \{5\}, \{6\}, \{7\}\}$ and $\{5\} \subseteq X_S$, we know that $\imath' \in I_{\rm bad}$. This is why transition $(a, \{e, a\})$ is not added to T_1 at state S_7 according to line 26; the same reason for state S_{10}. Then, we execute the while-loop in lines 5–7. Specifically, for state S_7, we have $\Sigma_{T_1}^{\rm succ}(S_7) = \{a\} \neq \Sigma_{\mathbb{T}}^{\rm succ}(S_7) = \{e, a\}$, i.e., feasible observable event e is not defined at this state. Therefore, state S_7 needs to be removed. Similarly, we need to remove state S_{10} for the same reason and the while-loop terminates. This results in T (accessible part) shown in the box marked with red-dashed lines in Fig. 2; for the sake of clarity, we denote this intermediate automaton by T_2. Note that, for state S_3 in T_2, although the transition to state S_7 is removed, it does not violate observation consistency since $(a, \{a\})$ is defined and event e is not a feasible observation at this state. Then, we proceed to lines 11–19 to find a decision deterministic subautomaton of T_2. Initially, we choose initial state S_1 and remove S_2 from initial states, since $\{a\} = O(S_2) \subset O(S_2) = \{e, a\}$. Then, at state S_1, for event $e \in \Sigma_T^{\rm succ}(S_1)$, we choose transition $(e, \{e, a\})$ and remove $(e, \{a\})$; for event $a \in \Sigma_T^{\rm succ}(S_1)$, the choice is unique, i.e., $(a, \{e, a\})$. Then, for states S_3, S_4, S_5, and S_6, only event a is feasible and at each state the sensing decision associated with a is unique. This gives automaton T shown as the red highlighted part in Fig. 2. Finally, by adding self-loops and renaming event names, we obtain dynamic mask $\Omega = (T, \Theta)$ shown in Fig. 3, which is a solution to Problem 1.

Remark 2: At state S_2 in Fig. 2, event $(a, \{a\})$ should also be defined according to the updating rule. However, this sensing decision is equivalent to $(a, \{e, a\})$ at this state since event e is redundant in the sense that it is not feasible within the unobservable reach encountered, i.e., e cannot be observed from $\{1, 5\}$ even if we choose to monitor it. Hence, taking $(a, \{a\})$ from S_2 will reach an information state that has exactly the same future behavior as state S_3. Therefore, for the sake of simplicity, we only depict one transition with all redundant events included instead of depicting all equivalent decisions.

Let us discuss the computational complexity of Algorithm 1. In the worst case, automaton T contains at most $2^{|X|} \cdot 2^{2^{|X| \times |X|}} \cdot 2^{|\Sigma_s|}$ states and $|\Sigma| \cdot 2^{|X|} \cdot 2^{2^{|X| \times |X|}} \cdot 4^{|\Sigma_s|}$ transitions. The running time of procedure EXPAND is linear in the number of transitions in T and the running time of the while-loop is quadratic

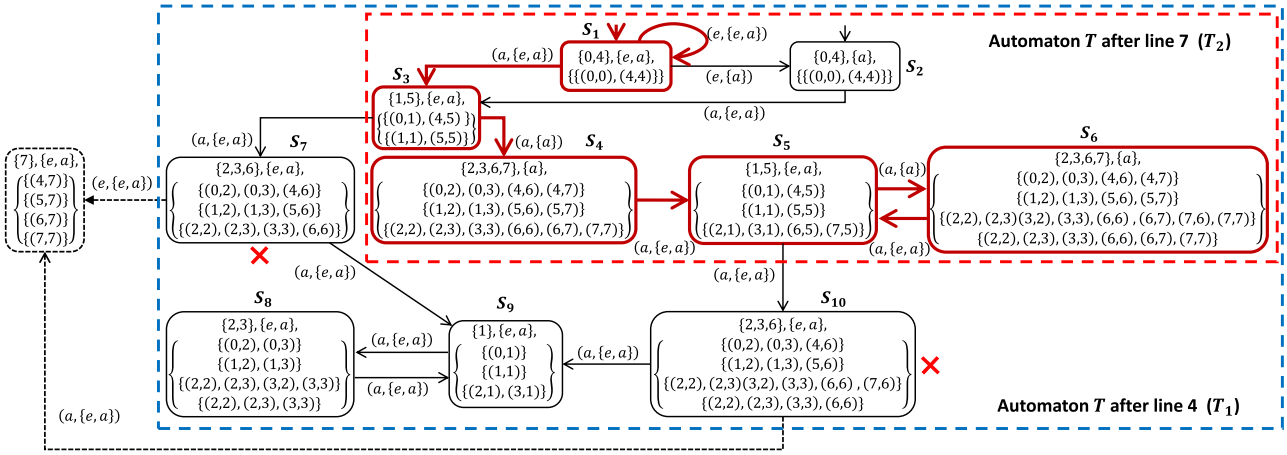

Fig. 2. For the sake of clarity, we put the sensing decision as the second component of each state in the figure.

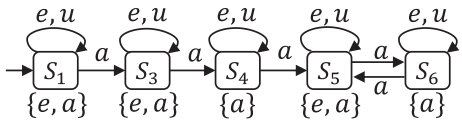

Fig. 3. Solution $\Omega = (T, \Theta)$.

in the number of states in T. Therefore, the overall complexity is doubly exponential in the number of states of G and exponential in the number of events. For current-state opacity, only exponential complexity is required for the synthesis of dynamic masks; see, e.g., [8]. Our higher complexity comes from the fact that infinite-step opacity is fundamentally more difficult than current-state opacity and delayed information is involved in this problem.

B. Correctness of the Synthesis Algorithm

In this section, we show that Algorithm 1 correctly solves Problem 1, in the sense that

1) Algorithm 1 is sound, i.e., any solution returned by Algorithm 1 is indeed maximal and infinite-step opaque; and
2) Algorithm 1 is complete, i.e., it will not return "no solution" when a solution to Problem 1 exists.

Throughout this section, we use $\Omega = (T, \Theta)$ to denote the dynamic mask returned by Algorithm 1.

First, we show the soundness of Algorithm 1.

Lemma 1: G is infinite-step opaque w.r.t. X_S and Ω. Moreover, for any Ω' such that G is infinite-step opaque w.r.t. X_S and Ω', we have $\Omega \not< \Omega'$.

Proof: Let $\alpha\beta \in P_\Omega(\mathcal{L}(G))$ be any observable string under Ω. By Corollary 1, we know that $\hat{X}_\Omega(\alpha \mid \alpha\beta) \in D_1(\mathcal{I}_\Omega(\alpha))$. By line 26 in Algorithm 1, we know that $\mathcal{I}_\Omega(\alpha) \notin I_{\text{bad}}$, which implies that $\hat{X}_\Omega(\alpha \mid \alpha\beta) \not\subseteq X_S$. Since $\alpha\beta$ is an arbitrary string, we know that Ω is infinite-step opaque.

To see the second point, we assume, for the sake of contradiction, that there exists Ω' such that 1) G is infinite-step opaque w.r.t. X_S and Ω'; and 2) $\Omega < \Omega'$. Then, let us define a new subautomaton of $\mathbb{T}$ denoted by $T_{\Omega'} = (A_{\Omega'}, \Sigma_{\Omega'}, \delta_{\Omega'}, A_{0,\Omega'})$ as follows:

1) $A_{\Omega'} = \{\mathcal{I}_{\Omega'}(\alpha) \in I : \alpha \in P_{\Omega'}(\mathcal{L}(G))\}$ is the set of states;
2) $A_{0,\Omega'} = \{(UR_{\Omega'(\epsilon)}(X_0), \{\odot_{\Omega'(\epsilon)}(UR_{\Omega'(\epsilon)}(X_0))\}, \Omega'(\epsilon))\}$, i.e., the initial state is unique;
3) The transition function $\delta_{\Omega'}$ is defined by: for any $\imath \in A_{\Omega'}, (\sigma, \theta) \in (\Sigma_o \cup \Sigma_s) \times \mathcal{O}$, $\delta_{\Omega'}(\imath, (\sigma, \theta))$ is defined if there exists $s\sigma \in \mathcal{L}(G)$ such that $\mathcal{I}_{\Omega'}(P_{\Omega'}(s)) = \imath, \sigma \in \Omega'(s)$ and $\theta = \Omega'(s\sigma)$.

Since Ω' is infinite-step opaque, by Corollary 1, we know that $T_{\Omega'}$ is safe. Therefore, all states and transitions in $T_{\Omega'}$ will be included in T after line 4 in Algorithm 1. Moreover, $T_{\Omega'}$ is observation consistent since transitions are defined for all feasible events. Therefore, we know that no state or transition is removed in the while-loop in Algorithm 1. Hence, all states and transitions in $T_{\Omega'}$ still remain in T after line 7 in Algorithm 1.

Now, since $\Omega < \Omega'$, we know that $\forall s \in \mathcal{L}(G) : \Omega(s) \subseteq \Omega'(s)$; and $\exists t \in \mathcal{L}(G) : \Omega(t) \subset \Omega'(t)$. Therefore, let us consider a string $t \in \mathcal{L}(G)$ such that $\Omega(t) \subset \Omega'(t)$ and $\forall s \in \overline{\{t\}} \setminus \{t\} : \Omega(s) = \Omega'(s)$. Let us consider the following two cases. If $t = \epsilon$, then we know that $\imath_0' := ((UR_{\Omega'(\epsilon)}(X_0), \{\odot_{\Omega'(\epsilon)}(UR_{\Omega'(\epsilon)}(X_0))\}, \Omega'(\epsilon))) \in A_{0,T}$ and $\Omega(\epsilon) = O(\imath_0) \subset O(\imath_0') = \Omega'(\epsilon)$, where $\imath_0$ is the initial state chosen in line 11 of Algorithm 1. However, this is a contradiction since we should choose at least $\imath_0'$ in this case. If $t \neq \epsilon$, then we write $t = t'\sigma$, where $\sigma \in \Omega(t')$. Since $\forall s \in \overline{\{t\}} \setminus \{t\} : \Omega(s) = \Omega'(s)$, we know that $P_\Omega(t') = P_{\Omega'}(t') =: \alpha$ and $\mathcal{I}_\Omega(\alpha) = \mathcal{I}_{\Omega'}(\alpha)$. Therefore, event $(\sigma, \Omega'(t))$ is defined at $\mathcal{I}_\Omega(\alpha)$ in T. However, this contradicts to our choice in line 15 as $\Omega(t) \subset \Omega'(t)$.

Next, we show the completeness of Algorithm 1. ■

Lemma 2: Algorithm 1 will not return "no solution" when a solution to Problem 1 exists.

Proof: Suppose that there exists a dynamic mask Ω' that solves Problem 1. Then, we define a new subautomaton of $\mathbb{T}$ denoted by $T_{\Omega'}$ by the same construction in the proof of Lemma 2. As we discussed in the proof of Lemma 2, all states and transitions in $T_{\Omega'}$ still remain in T after line 7 in

Algorithm 1. Therefore, $A_{T,0}$ is not empty and line 8 will not return "no solution." ■

Finally, we summarize Lemmas 1 and 2 by the following theorem.

Theorem 1: Algorithm 1 correctly solves Problem 1.

Remark 3: In the proof of Lemma 1, we show that, for any infinite-step opaque dynamic mask Ω', its corresponding $T_{\Omega'}$ is "included" in T after line 7 in Algorithm 1. Therefore, T after line 7 in Algorithm 1 essentially "embeds" all possible solutions in it. Recall that, in this paper, we aim to find one logical maximal dynamic mask for infinite-step opacity. Another interesting question is how to compare two incomparable maximal solutions. This may require us to introduce new numerical measure for optimality, e.g., quantitative cost. This direction has actually been explored in [8] for current-state opacity by solving a mean payoff game. In our context, one could also apply similar techniques over the structure obtained after line 7 in Algorithm 1. This numerical optimization problem is beyond the scope of this paper as the main purpose of this paper is to show what is the right information structure to handle delayed information in infinite-step opacity.

V. Complexity Reduction and Generalization

In this section, we first present an approach to reduce the state space of information states for infinite-step opacity. Then, we discuss how to generalize the proposed synthesis algorithm to other properties with delayed information.

A. Information-State Reduction

In the previous sections, we have provided an approach for synthesizing dynamic masks that preserve infinite-step opacity. The idea is to explore the information state space $2^X \times 2^{2^{X\times X}} \times \mathcal{O}$ following the proposed information state updating rule. Nevertheless, this is still a very large state space; the main complexity comes from the fact that we need to store *all possible* delayed state estimates for each instant in order to check whether or not the current information reveals some secret in the past. For the purpose of preserving infinite-step opacity, however, not all delayed state estimates along the decision history are needed. It is possible that a current/delayed state estimate of an instant does not contain a secret state. This may correspond to the following two cases: 1) the current-state estimate for that instant does not even contain a secret state; or 2) the possibility that the system was in a secret state for that instant, which has been "smoothed" by the future information. Then, for such a scenario, no matter what information is released in the future, such an instant will not be a source that reveals the secret. Therefore, we do not need to store state estimate and its induced delayed state estimates in the future. This observation allows us to further reduce the state space of the information state.

Formally, we define

$$\Xi_S = \{\rho \in 2^{X\times X} : \exists (x,x') \in \rho \text{ s.t. } x \in X_S\}$$

as the set of all state-pair-sets in which the first component of some state pairs is a secret state. Then, we modified the information state updating rule as follows. For each information state $\imath = (C(\imath), D(\imath), O(\imath)) \in 2^X \times 2^{2^{X\times X}} \times \mathcal{O}$, suppose a monitored event $\sigma \in O(\imath)$ is observed and a new sensing decision $\theta \in \mathcal{O}$ is made immediately after the occurrence of σ. Then, the new information state reached $\imath' = (C(\imath'), D(\imath'), O(\imath'))$ is defined by

$$\begin{cases} C(\imath') = & UR_\theta(NX_\sigma(C(\imath))) \\ D(\imath') = & \left(\{\widetilde{UR}_\theta(\widetilde{NX}_\sigma(\rho)) \in 2^{X\times X} : \rho \in D(\imath)\} \cap \Xi_S\right) \\ & \cup \left(\{\odot_\theta(C(\imath'))\} \cap \Xi_S\right) \\ O(\imath') = & \theta \end{cases}. \tag{11}$$

Equations (11) is also referred to as the *reduced information state updating rule*. The difference between the reduced updating rule and the original updating rule is the second component $D(\imath')$. Specifically, in the original updating rule, we need to update all previous delayed state estimates and always add the current state estimate (after mapping) $\odot_\theta(C(\imath'))$ to this component. However, in the reduced updating rule, we only need to update those delayed state estimates in which the secret behaviors are still possible. Note that some secret states contained in previous delayed state estimates may be smoothed by the new information. Therefore, we can just forget those delayed state estimates for which the possibility of secret states has been eliminated. Similarly, we will only add the current state estimate when it contains a secret state. Otherwise, the current instant will never be a source that violates infinite-step opacity in the future and we do not need to remember this information.

In order to synthesize a dynamic mask that preserves infinite-step opacity, we can then use exactly the same algorithm, i.e., Algorithm 1, except the following differences.

1) The initial states $A_{0,T}$ in lines 1 should be replaced by
$$\{(UR_\theta(X_0), \{\odot_\theta(UR_\theta(X_0))\} \cap \Xi_S, \theta) \in X_T : \theta \in \mathcal{O}\} \setminus I_{\text{bad}}.$$
2) The transition function $\delta_{\mathbb{T}}$ considered in line 25 should follow the reduced information state updating rule, as defined in (11), not the originally one.

Compared with the original information state updating rule that may explore the entire $2^X \times 2^{2^{X\times X}} \times \mathcal{O}$ state space, the reduced information state updating rule will only explore a subspace

$$I_{\text{reduc}} := 2^X \times (2^{2^{X\times X}} \setminus 2^{2^{(X\setminus X_S)\times X}}) \times \mathcal{O}$$

that contains at most $2^{|X|+|\Sigma_s|} \cdot (2^{2^{|X|\times|X|}} - 2^{2^{|X\setminus X_S|\times|X|}})$ states. Therefore, the complexity of the synthesis algorithm can be reduced considerably when the number of secret states is relatively smaller (which is usually the case for most of the applications).

Next, we illustrate how to synthesize a dynamic mask using the reduced information state updating rule.

Example 6: Let us still consider system G in Fig. 1(a) with $\Sigma_o = \{a\}, \Sigma_{uo} = \{u\}, \Sigma_s = \{e\}$ and $X_S = \{5\}$. We apply Algorithm 1 using the reduce information state updating rule. The synthesis procedure is depicted in Fig. 5. For example, we consider that the initial state of T is $(\{0,4\}, \{\emptyset\}, \{e,a\})$ not the original one $(\{0,4\}, \{\{(0,0),(4,4)\}\}, \{e,a\})$; this

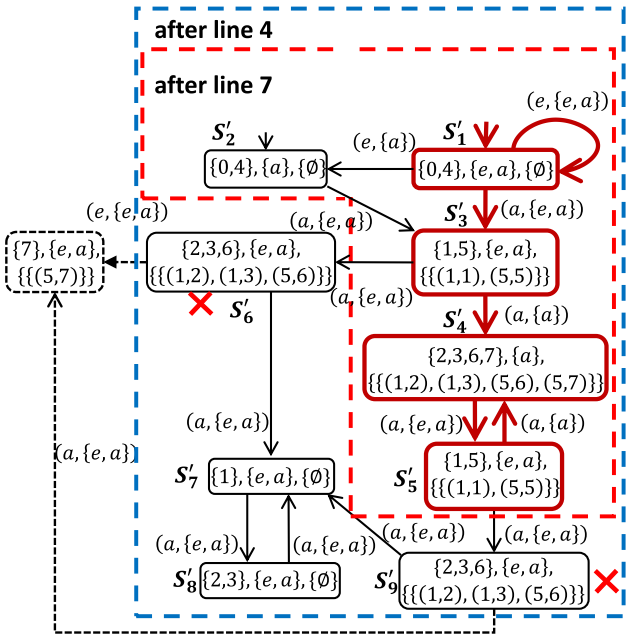

Fig. 4. Illustration of Algorithm 1 using the reduced information state updating rule.

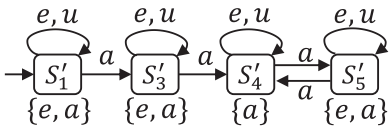

Fig. 5. $\Omega' = (T', \Theta')$ obtained based on the reduced information state.

is because $\{(0,0),(4,4)\} \notin \Xi_S$. From state S'_1 to S'_3, we need to add $\{\odot_{\{e,a\}}(\{1,5\})\}$ to the second component since $\{(1,1),(5,5)\} \in \Xi_S$. However, from state S'_3 to S'_4, we do not need to add $\{\odot_{\{e,a\}}(\{2,3,6,7\})\}$ to the second component since $\{(2,2),(2,3),(3,3),(6,7),(6,7),(7,7)\} \notin \Xi_S$ and we just need to update the delayed state estimate from $\{(1,1),(5,5)\}$ to $\{(1,2),(1,3),(5,6),(5,7)\}$. Following the same synthesis procedure as we discussed in Example 5, we obtain automaton T' shown as the part highlight in red in Fig. 4. By adding self-loops and renaming event names, we obtain dynamic mask $\Omega' = (T', \Theta')$ shown in Fig. 5.

Remark 4: By comparing $\Omega' = (T', \Theta')$ in Fig. 5 with $\Omega = (T, \Theta)$ in Fig. 3, we see that these two dynamic masks work essentially the same, but T' contains less states than T. This illustrates that some states in the dynamic mask synthesized (even using the reduced information state updating rule) may be redundant. For example, in $\Omega' = (T', \Theta')$ shown in Fig. 5, since states S'_3 and S'_5 are equivalent states in the sense that they have the same current decision and future behaviors. This dynamic mask can then be further reduced to $\Omega'' = (T'', \Theta'')$ shown in Fig. 6 by merging states S'_3 and S'_5 without changing its functionality. Therefore, after the execution of Algorithm 1, one can apply the standard automaton minimization algorithm, which can be done in polynomial-time for deterministic finite-state automata,

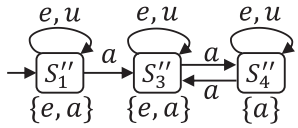

Fig. 6. $\Omega'' = (T'', \Theta'')$ with a minimal state space.

to further reduce the state space of the synthesized dynamic mask in order to save memory for its online implementation.

B. Synthesis for Initial-State Opacity

The proposed synthesis procedure and the reduction rule can also be easily extended to initial-state opacity. Specifically, system G is said to be *initial-state opaque* w.r.t. X_S and Ω if the intruder can never know for sure that the system is initially from a secret state, i.e.,

$$\forall \alpha \in P_\Omega(\mathcal{L}(G)) : \hat{X}_\Omega(\epsilon \mid \alpha) \not\subseteq X_S.$$

Similarly, to synthesize a dynamic mask that preserves initial-state opacity, we can still use Algorithm 1 with the following modifications.

1) The initial states $A_{0,T}$ in lines 1 is also replaced by

$$\{(UR_\theta(X_0), \{\odot_\theta(UR_\theta(X_0))\} \cap \Xi_S, \theta) \in X_T : \theta \in \mathcal{O}\} \setminus I_{\text{bad}}.$$

2) The second component of the transition function $\delta_{\mathbb{T}}$ considered in line 25, i.e., the part for $D(\imath)$, is replaced by

$$D(\imath') = \{\widetilde{UR}_\theta(\widetilde{NX}_\sigma(\rho)) \in 2^{X \times X} : \rho \in D(\imath)\} \cap \Xi_S.$$

Intuitively, the above modification says that, when we update the information state, we will only update the delayed-state-estimate that corresponds to the initial state and do not need to add any current-state estimate encountered. Therefore, for initial-state opacity, we only need to explore a subspace $2^X \times (2^{X \times X} \setminus 2^{X_S \times X}) \times \mathcal{O}$ as $D(\imath)$ is always a singleton. Hence, the complexity for the case of initial-state opacity is just single exponential.

Note that, although it has been show by [41] that initial-state opacity and current-state opacity can be mapped from one to the other for the purpose of verification, it is not the case for the purpose of synthesis. To the best of our knowledge, how to synthesize a dynamic mask for initial-state opacity has never been solved in the literature. This problem can now be solved within our framework.

C. Generalization to a Class of Properties

In fact, the proposed information state and its updating rule can be applied to the synthesis problems for other properties where delayed information is involved.

For example, in the computer science literature, K-anonymity is a property requiring that the observer should never know the current state of the system "too precisely" in the sense that the cardinality of the current-state estimate should always be larger than or equal to K, i.e.,

$$\forall \alpha \in P_\Omega(\mathcal{L}(G)) : |\hat{X}_\Omega(\alpha)| \geq K.$$

Note that the standard definition of K-anonymity only considers current information; it can be enforced using the approach in [47]. However, similar to the case of infinite-step opacity, the observer may also use future information to better know the state of the system for some previous instant. Therefore, we can define *infinite-step K-anonymity* as a stronger condition that takes the effect of future information into account. Specifically, system G is said to be *infinite-step K-anonymous* w.r.t. a positive integer $K \in \mathbb{N}$ if

$$\forall \alpha\beta \in P_\Omega(\mathcal{L}(G)) : |\hat{X}_\Omega(\alpha \mid \alpha\beta)| \geq K.$$

Intuitively, infinite-step K-anonymity says that the intruder should never know the state of the system at any specific instant "too precisely" in the sense that, for each instant, there always exist at least K distinct states that are indistinguishable even by using future information. We can also synthesize a dynamic mask that preserves infinite-step K-anonymity using the proposed approach. The only difference with infinite-step opacity is that, instead of considering illegal information states I_{bad} defined in (10), we need to consider the following illegal information states:

$$I_{\text{bad}}^{\text{ano}} = \{\imath \in I : \exists q \in D_1(\imath) \text{ s.t. } |q| < K\}.$$

According to Corollary 1, for each state in $I_{\text{bad}}^{\text{ano}}$, there must exist a previous instant in the trajectory reaching this information such that the observer knows for sure that the cardinality of the delayed state estimate is smaller than K, i.e., infinite-step K-anonymity is violated.

Another related property that can be enforced by the proposed approach is *infinite-step indistinguishability*. Specifically, let $X_1, X_2 \subseteq X$ be two disjoint sets of states. Then, infinite-step indistinguishability requires that the intruder should never be able to distinguish X_1 and X_2 for any specific instant, i.e.,

$$(\forall \alpha\beta \in P_\Omega(\mathcal{L}(G)))(\forall i, j \in \{1,2\} : i \neq j)$$

$$\hat{X}_\Omega(\alpha \mid \alpha\beta) \cap X_i \neq \emptyset \Rightarrow \hat{X}_\Omega(\alpha \mid \alpha\beta) \cap X_j \neq \emptyset.$$

Similarly, we can synthesize a dynamic mask that preserves infinite-step indistinguishability by replacing the original illegal information states I_{bad} as

$$I_{\text{bad}}^{\text{indis}} = \left\{ \imath \in I : \begin{array}{c} \exists q \in D_1(\imath), \exists i, j \in \{1,2\} \text{ s.t.} \\ q \cap X_i \neq \emptyset \wedge q \cap X_j = \emptyset \end{array} \right\}.$$

In fact, our synthesis algorithm can be applied to any property that can be written as a safety-type predicate $\varphi : 2^X \times 2^{2^X} \to \{0,1\}$, where first part represents the current-state estimate of the system and the second part represents all possible delayed-state estimates of the system.

Example 7: Let us still consider system G in Fig. 1(a) with $\Sigma_o = \{a\}, \Sigma_{uo} = \{u\}$ and $\Sigma_s = \{e\}$. However, instead of considering infinite-step opacity, we consider infinite-step K-anonymity for $K = 2$. Then, for the structure in Fig. 2, we also have $\imath = (\{7\}, \{e, a\}, \{\{(4,7)\}, \{(5,7)\}, \{(6,7)\}, \{(7,7)\}\}) \in I_{\text{bad}}^{\text{ano}}$ since $\{4\} \in D_1(\imath)$ but $|\{4\}| = 1 < 2$. Therefore, both states S_7 and S_{10} still need to be violated. Then, the remaining part of the synthesis procedure is exactly the same as Example 5 and we also obtain dynamic mask $\Omega = (T, \Theta)$ shown in Fig. 3 but for infinite-step 2-anonymity.

VI. Conclusion

We investigated the problem of synthesizing dynamic masks that preserve infinite-step opacity while maximizing the information acquired. A new type of information state was proposed to capture all delayed information for all previous instants. An effective algorithm was provided to solve the synthesis problem. Our result extends previous work on the synthesis of dynamic masks from current-state opacity to infinite-step opacity. We believe that the new type of information state proposed in this paper can also be applied to the supervisor synthesis problem and the insertion function synthesis problem for the purpose of enforcing infinite-step opacity. We will explore these directions in the future.

References

[1] E. Badouel, M. Bednarczyk, A. Borzyszkowski, B. Caillaud, and P. Darondeau, "Concurrent secrets," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 17, no. 4, pp. 425–446, 2007.

[2] F. Basile and G. De Tommasi, "An algebraic characterization of language-based opacity in labeled Petri nets," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 340–347.

[3] B. Bérard, J. Mullins, and M. Sassolas, "Quantifying opacity," *Math. Struct. Comput. Sci.*, vol. 25, no. 2, pp. 361–403, 2015.

[4] A. Bourouis, K. Klai, N. Ben Hadj-Alouane, and Y. El Touati, "On the verification of opacity in web services and their composition," *IEEE Trans. Services Comput.*, vol. 10, no. 1, pp. 66–79, Jan./Feb. 2017.

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

[6] J. W. Bryans, M. Koutny, and P. Ryan, "Modelling opacity using Petri nets," *Electron. Notes Theor. Comput. Sci.*, vol. 121, pp. 101–115, 2005.

[7] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. New York, NY, USA: Springer, 2008.

[8] F. Cassez, J. Dubreil, and H. Marchand, "Synthesis of opaque systems with static and dynamic masks," *Formal Methods Syst. Des.*, vol. 40, no. 1, pp. 88–115, 2012.

[9] F. Cassez and S. Tripakis, "Fault diagnosis with static and dynamic observers," *Fundamenta Informaticae*, vol. 88, no. 4, pp. 497–540, 2008.

[10] S. Chédor, C. Morvan, S. Pinchinat, and H. Marchand, "Diagnosis and opacity problems for infinite state systems modeled by recursive tile systems," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 25, nos. 1/2, pp. 271–294, 2015.

[11] J. Chen, M. Ibrahim, and R. Kumar, "Quantification of secrecy in partially observed stochastic discrete event systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 185–195, Jan. 2016.

[12] X. Cong, M. P. Fanti, A. M. Mangini, and Z. Li, "On-line verification of current-state opacity by petri nets and integer linear programming," *Automatica*, vol. 94, pp. 205–213, 2018.

[13] E. Dallal and S. Lafortune, "On most permissive observers in dynamic sensor activation problems," *IEEE Trans. Autom. Control*, vol. 59, no. 4, pp. 966–981, Apr. 2014.

[14] P. Darondeau, H. Marchand, and L. Ricker, "Enforcing opacity of regular predicates on modal transition systems," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 25, nos. 1/2, pp. 251–270, 2015.

[15] J. Dubreil, P. Darondeau, and H. Marchand, "Supervisory control for opacity," *IEEE Trans. Autom. Control*, vol. 55, no. 5, pp. 1089–1100, May 2010.

[16] Y. Falcone and H. Marchand, "Enforcement and validation (at runtime) of various notions of opacity," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 25, no. 4, pp. 1–40, 2014.

[17] L. Hélouët, H. Marchand, and L. Ricker, "Opacity with powerful attackers," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 475–482.

[18] R. Jacob, J.-J. Lesage, and J.-M. Faure, "Overview of discrete event systems opacity: Models, validation, and quantification," *Annu. Rev. Control*, vol. 41, pp. 135–146, 2016.

[19] Y. Ji, Y.-C. Wu, and S. Lafortune, "Enforcement of opacity by public and private insertion functions," *Automatica*, vol. 93, pp. 369–378, 2018.
[20] R. J. Barcelos and J. C. Basilio, "Enforcing current-state opacity through shuffle in event observations," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 106–111.
[21] C. Keroglou and C. N. Hadjicostis, "Probabilistic system opacity in discrete event systems," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 28, no. 2, pp. 289–314, 2018.
[22] C. Keroglou, L. Ricker, and S. Lafortune, "Insertion functions with memory for opacity enforcement," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 405–410.
[23] K. Kobayashi and K. Hiraishi, "Verification of opacity and diagnosability for pushdown systems," *J. Appl. Math.*, vol. 2013, 2013, Art. no. 654059.
[24] S. Lafortune, F. Lin, and C. N. Hadjicostis, "On the history of diagnosability and opacity in discrete event systems," *Annu. Rev. Control*, vol. 45, pp. 257–266, 2018.
[25] F. Lin, "Opacity of discrete event systems and its applications," *Automatica*, vol. 47, no. 3, pp. 496–503, 2011.
[26] J. Mullins and M. Yeddes, "Opacity with orwellian observers and intransitive non-interference," in *Proc. 12th Int. Workshop Discrete Event Syst.*, 2014, pp. 344–349.
[27] M. Noori-Hosseini, B. Lennartson, and C. Hadjicostis, "Compositional visible bisimulation abstraction applied to opacity verification," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 445–452.
[28] A. Saboori and C. N. Hadjicostis, "Coverage analysis of mobile agent trajectory via state-based opacity formulations," *Control Eng. Pract.*, vol. 19, no. 9, pp. 967–977, 2011.
[29] A. Saboori and C. N. Hadjicostis, "Verification of K-step opacity and analysis of its complexity," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 3, pp. 549–559, Jul. 2011.
[30] A. Saboori and C. N. Hadjicostis, "Opacity-enforcing supervisory strategies via state estimator constructions," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1155–1165, May 2012.
[31] 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.
[32] 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.
[33] D. Sears and K. Rudie, "Minimal sensor activation and minimal communication in discrete-event systems," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 26, no. 2, pp. 295–349, 2016.
[34] S. Shu, Z. Huang, and F. Lin, "Online sensor activation for detectability of discrete event systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 457–461, Apr. 2013.
[35] S. Takai and Y. Oka, "A formula for the supremal controllable and opaque sublanguage arising in supervisory control," *SICE J. Control, Measu. Syst. Integr.*, vol. 1, no. 4, pp. 307–311, 2008.
[36] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Decidability of opacity verification problems in labeled Petri net systems," *Automatica*, vol. 80, pp. 48–53, 2017.
[37] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Verification of state-based opacity using Petri nets," *IEEE Trans. Autom. Control*, vol. 62, no. 6, pp. 2823–2837, Jun. 2017.
[38] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Current-state opacity enforcement in discrete event systems under incomparable observations," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 28, no. 2, pp. 161–182, 2018.
[39] B. Wu, J. Dai, and H. Lin, "Synthesis of insertion functions to enforce decentralized and joint opacity properties of discrete-event systems," in *Proc. Am. Control Conf.*, 2018, pp. 3026–3031.
[40] B. Wu, Z. Liu, and H. Lin, "Parameter and insertion function co-synthesis for opacity enhancement in parametric stochastic discrete event systems," in *Proc. Annu. Am. Control Conf.*, 2018, pp. 3032–3037.
[41] Y.-C. Wu and S. Lafortune, "Comparative analysis of related notions of opacity in centralized and coordinated architectures," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 23, no. 3, pp. 307–339, 2013.
[42] Y.-C. Wu and S. Lafortune, "Synthesis of insertion functions for enforcement of opacity security properties," *Automatica*, vol. 50, no. 5, pp. 1336–1348, 2014.
[43] Y.-C. Wu, K. A. Sankararaman, and S. Lafortune, "Ensuring privacy in location-based services: An approach based on opacity enforcement," in *Proc. 12th Int. Workshop Discrete Event Syst.*, 2014, pp. 33–38.
[44] B. Xing, J. Dai, and S. Liu, "Enforcement of opacity security properties for ship information system," *Int. J. Naval Archit. Ocean Eng.*, vol. 8, no. 5, pp. 423–433, 2016.
[45] 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.
[46] X. Yin and S. Lafortune, "A new approach for the verification of infinite-step and K-step opacity using two-way observers," *Automatica*, vol. 80, pp. 162–171, 2017.
[47] X. Yin and S. Lafortune, "A general approach for optimizing dynamic sensor activation for discrete event systems," *Automatica*, vol. 105, pp. 376–383, 2019.
[48] X. Yin and S. Li, "Synthesis of dynamic masks for infinite-step opacity," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 354–359.
[49] X. Yin, Z. Li, W. Wang, and S. Li, "Infinite-step opacity and K-step opacity of stochastic discrete-event systems," *Automatica*, vol. 99, pp. 266–274, 2019.
[50] B. Zhang, S. Shu, and F. Lin, "Maximum information release while ensuring opacity in discrete event systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 4, pp. 1067–1079, Jul. 2015.
[51] K. Zhang, X. Yin, and M. Zamani, "Opacity of nondeterministic transition systems: A (bi) simulation relation approach," *IEEE Trans. Autom. Control*, doi: 10.1109/TAC.2019.2908726.

Xiang Yin (M'17) was born in Anhui, China, in 1991. He received the B.Eng. degree from Zhejiang University, Hangzhou, China, in 2012, and the M.S. and Ph.D. degrees from the University of Michigan, Ann Arbor, Ann Arbor, MI, USA, in 2013 and in 2017, respectively, all in electrical engineering.

Since 2017, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is an Associate Professor. His research interests include formal methods, control of discrete-event systems, model-based fault diagnosis, security, and their applications to cyber and cyber-physical systems.

Dr. Yin received the Outstanding Reviewer Awards from *Automatica*, the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, and the *Journal of Discrete Event Dynamic Systems*. He also received the IEEE Conference on Decision and Control Best Student Paper Award Finalist in 2016. He is the Co-Chair of the IEEE CSS Technical Committee on Discrete Event Systems.

Shaoyuan Li (SM'05) was born in Hebei, China, in 1965. He received the B.S. and M.S. degrees in automation from the Hebei University of Technology, Tianjin, China, in 1987 and 1992, respectively, and the Ph.D. degree from Nankai University, Tianjin, China, in 1997.

Since 1997, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is currently a Professor. His current research interests include model predictive control, dynamic system optimization, and cyber-physical systems. He is the Vice-President of the Chinese Association of Automation.