arXiv:1802.05728v1 [cs.CR] 15 Feb 2018

Synthesis of Insertion Functions to Enforce Decentralized and Joint Opacity Properties of Discrete-event Systems

Bo Wu, Jin Dai and Hai Lin

***Abstract*— Opacity is a confidentiality property that characterizes the non-disclosure of specified secret information of a system to an outside observer. In this paper, we consider the enforcement of opacity within the discrete-event system formalism in the presence of multiple intruders. We study two cases, one without coordination among the intruders and the other with coordination. We propose appropriate notions of opacity corresponding to the two cases, respectively, and propose enforcement mechanisms for these opacity properties based on the implementation of insertion functions, which manipulates the output of the system by inserting fictitious observable events whenever necessary. The insertion mechanism is adapted to the decentralized framework to enforce opacity when no coordination exists. Furthermore, we present a coordination and refinement procedure to synthesize appropriate insertion functions to enforce opacity when intruders may coordinate with each other by following an intersection-based coordination protocol. The effectiveness of the proposed opacity-enforcement approaches is validated through illustrative examples.**

I. Introduction

Security and privacy have become important issues in the design of cyber and cyber-physical systems [1]. In this paper, we focus our study on *opacity* [2], which is a confidentiality property that justifies whether a given system's confidential information (denoted as *"secret"*) is kept uncertain from an external observer (termed as an *intruder*). Since many security and privacy properties, such as anonymity [3], trace-based non-interference [4] and secrecy [5], [6], can be expressed in terms of opacity [7], it has emerged as an active research topic in the computer science and control literature, see, e.g., [8] and the references therein.

Motivated by the fact that many engineering systems are inherently event-driven, we consider opacity issues in the framework of discrete-event systems (DES) [9]. An opacity problem is generally formulated as follows in the context of DES: (i) the system is modeled as a Petri net [10], [11] or a finite automaton [12]; (ii) the system possesses a secret that is expected to be hidden from an intruder; (iii) the intruder is an observer with full knowledge of the system's structure but can only observe part of the system's behavior. The system is said to be opaque with respect to the given secret if the intruder can never determine unambiguously that the secret has occurred based on its observation of the system's behaviors. More specifically, if opacity of the system holds, then for any behavior that may reveal the secret (termed *secret behavior*), there exists at least one behavior that does not reveal the secret (termed *non-secret behavior*) which shares the same observation of the secret behavior to the intruder; thus, the intruder can never be sure if the secret or the non-secret has occurred. Depending on how the secret is represented, various notions of opacity have been introduced in the literature, and considerable amount of research efforts has been devoted to the formal verification of language-based opacity [7], current-state opacity [13], initial-state opacity [12], K-step opacity [14] and infinite-step opacity [15].

In case the system fails to be opaque, formal methods have also been proposed to enforce opacity. Design of opacity-enforcing supervisory controllers for restricting the system's behavior to ensure opacity by disabling any behavior that will reveal the secret has been studied extensively in literature [16]–[19]. Nevertheless, the supervisory control approach is not suitable for situations where the system must execute its full behavior. A runtime mechanism was developed in [20] to enforce K-step opacity based on delaying the output; however, this method only ensured opacity of secrets whose time duration was of concern. Rather than supervisory control approach, we consider enforcement strategies that do not alter the behavior of the system and instead ensure opacity by appropriately manipulating the system's output information whenever necessary. One of the enforcement techniques was implemented by a dynamic observer in [21]. However, the intermittent loss of observability of certain events may render the observation of the intruder inconsistent with its knowledge of the system and its original observation capabilities, which may remind the intruder of the existence of the opacity-enforcement mechanism. Wu and Lafortune [22] proposed an enforcement mechanism based on insertion of fictitious observable events at the system's output; the inserted events were observationally equivalent to the system's genuine observable events from the intruder's perspective, therefore making the intruder confused.

Recent advances in communication and network technologies have made large-scale systems with spatially-decentralized and/or distributed architectures more widely used in the application; therefore, opacity problems for DES with decentralized structure are of both academic and practical importance. For instance, for a cryptosystem that can be observed by users of multiple security levels, opacity should be guaranteed in such a way that: (i) users with lower security level can never infer any information which can only be accessed by users of high security level [4]; (ii) even if a user has a high security level, it is still not able to infer any private information that is possessed by a user with low

The partial support of the National Science Foundation (Grant No. CNS-1446288, ECCS-1253488, IIS-1724070) and of the Army Research Laboratory (Grant No. W911NF- 17-1-0072) is gratefully acknowledged.

The authors are with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN, 46556 USA. bwu3@nd.edu, jdai1@nd.edu, hlin1@nd.edu

security level [23]. Compared to the fruitful contributions that have been made to opacity problems in the presence of a single intruder, limited studies have been made to the cases where the system can be observed by multiple intruders. Badouel et al. [23] considered multiple intruders, each of them having its own observation mapping and the secret of interest. The system therein was said to be concurrently opaque if all secrets can be kept safe. A different notion termed as "joint opacity" was proposed in [24], in which a team of intruders collaborated through a coordinator to infer the secret of common interest. Paoli and Lin [25] studied decentralized opacity issues with and without coordination among the intruders. Nevertheless, to the best of the authors' knowledge, most of the existing results are established on opacity verification problems while no prior work has been proposed to investigate opacity-enforcement problems in the presence of multiple intruders.

We are therefore motivated to study opacity-enforcement problems of DES that can be observed by multiple intruders. By modeling the system as a finite automaton, we assume that each intruder has full prior knowledge of the system model but can only partially observe the behavior of the system. We investigate opacity problems in two cases, one assuming no coordination among the intruders and the other assuming that the intruders may coordinate with each other. We adopt the enforcement mechanism based on insertion functions to assure decentralized opacity when no coordination exists among the intruders. Furthermore, we study the enforcement of joint opacity when the intruders may coordinate via an intersection-based protocol. Facing the coordinated intruders, we propose a centralized coordination and refinement procedure to construct local insertion functions associated with each intruder's observation capabilities such that joint opacity can be guaranteed.

The remainder of this paper is organized as follows. We present the system model and relevant concepts of opacity problems in DES and the insertion-based opacity-enforcement mechanism of DES in Section II. We study the opacity-enforcement problem of DES in the presence of multiple non-coordinating intruders and compute appropriate insertion functions for each intruder in Section III. Under the assumption that the intruders may coordinate via an intersection-based protocol, we introduce the notion of joint opacity in Section IV and develop enforcement schemes for joint opacity by incorporating the synthesis of local insertion functions with centralized coordination. Finally, we end this paper with concluding remarks and discussion of future research directions in Section V.

II. Opacity of Discrete-event Systems

A. Preliminaries of Discrete-event Systems

The following notation and concepts are standard in the DES literature [9]. For a finite alphabet E of event symbols, $|E|$ and 2^E denote the cardinality and power set of E, respectively. E^* stands for the set of all finite strings over E plus the empty string ϵ. A subset of $L \subseteq E^*$ is called a language over E. The prefix closure of L is defined by $\overline{L} = \{s \in E^* | (\exists t \in E^*)[st \in L]\}$. L is said to be *prefix-closed* if $L = \overline{L}$.

We consider the DES modeled as a *non-deterministic finite automaton* (NFA) $G = (X, E, f, X_0)$, where X is the finite set of states, E is the finite set of events, $f : X \times E \to 2^X$ is the (partial) transition function, $X_0 \subseteq X$ is the set of initial states. The transition function f can be extended to $X \times E^*$ in the natural way [9].Given a set $X' \subseteq X$ of states, the language generated by G from X' is defined by $L(G, X') = \{s \in E^* | (\exists x' \in X')[f(x', s)!]\}$, where $f(x', s)!$ means that the transition $f(x', s)$ is defined. The generated behavior of G is then given by $L(G, X_0)$. We write $L(G)$ for simplicity if X_0 is clear from the context.

In general, the system G can only be partially observed. Towards this end, E is partitioned into two disjoint subsets, i.e., $E = E_o \dot{\cup} E_{uo}$, where E_o is the set of observable events and E_{uo} is the set of unobservable events. The presence of partial observation is captured by the natural projection $P : E^* \to E_o^*$, which is defined as:

$$P(\epsilon) = \epsilon, \text{ and } P(se) = \begin{cases} P(s)e, & \text{if } e \in E_o \\ P(s), & \text{if } e \in E_{uo} \end{cases} \quad (1)$$

for all $s \in E^*$ and $e \in E$. The inverse projection of P is defined as $P^{-1}(t) = \{s \in E^* | P(s) = P(t)\}$ for $t \in E_o^*$.

B. Current-state Opacity of Discrete-event Systems

The ingredients of an opacity-enforcement problem in DES include: (i) G has a secret; (ii) the intruder is an observer with full knowledge of the structure of G; (iii) the intruder can only observe the behavior of G partially due to its limited observation capabilities E_o. With the prior knowledge of G, the intruder can infer the system's evolution by constructing estimates on the basis of online observations. Depending on how the secret is defined, various notions of opacity have been extensively studied in the literature. In this paper, we define the secret to be a set of states of G and consider the notion of *current-state opacity*. Intuitively, the system G is current-state opaque if for any secret behavior that visits a secret state, there always exists a non-secret behavior of G that visits a non-secret state while the intruder cannot distinguish between these two behaviors. Formally, current-state opacity is defined as follows.

Definition 1 (Current-state Opacity (CSO)): Given the set of observable events $E_o \subseteq E$, the set of *secret states* $X_S \subseteq X$ and the set of *non-secret states* $X_{NS} \subseteq X$, the system $G = (X, E, f, X_0)$ is said to be current-state opaque with respect to E_o, X_S and X_{NS} if

$$(\forall x_0 \in X_0)(\forall t \in L(G, x_0) : f(x_0, t) \in X_S) \Rightarrow (\exists x_0' \in X_0)$$
$$(\exists t' \in L(G, x_0'))[(f(x_0', t') \in X_{NS}) \wedge (P(t') = P(t))]. \quad (2)$$

Remark 1: We assume without loss of generality in the rest of this paper that the set of non-secret states is the complement of the secret state set, i.e., $X_{NS} = X \setminus X_S$.

Remark 2: According to [24], other notions of opacity, specifically language-based opacity, initial-state opacity and initial-and-final-state opacity, can all be transformed to CSO

in polynomial time. Thus, our proposed enforcement approach for CSO of DES applies to the enforcement of other opacity notions as well.

C. Event Insertion Mechanism

In [22], the authors proposed an opacity-enforcement mechanism based on the implementation of insertion functions when the system G fails to be CSO. As shown in Fig. 1, an insertion function serves as a special monitoring interface between the system and the intruder. The insertion function receives an output behavior in $s \in P[L(G)]$ and inserts fictitious observable events before s is observed whenever the intruder may infer the occurrence of the secret from s. It is worth pointing out that the intruder cannot distinguish inserted observable events from the system's genuine observable events.

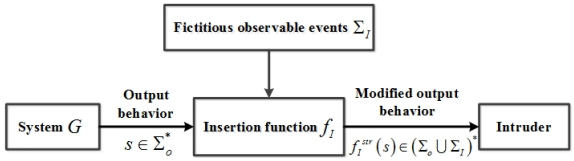


Fig. 1: Opacity-enforcement based on event insertion.

For the purpose of clear presentation, we associate each inserted event with an "insertion label" I, and the set of inserted events is denoted by $E_I = \{e_I : e \in E_o\}$. Formally, the basic structure of an insertion function is defined as a (possibly partial) mapping f_I: $E_o^* \times E_o \to E_I^* E_o$ that outputs a string with necessarily inserted events based on the system's historical and current output behavior. Given a string $se_o \in P[L(G)]$ that has been observed by the insertion function, the output behavior of the insertion function before the occurrence of e_o is defined as $f_I(s, e_o) = s_I e_o$ where $s_I \in E_I^*$ is the inserted string. In the sequel, we assume additionally that length of s_I is bounded from above. To determine the complete modified output from the insertion function, we define recursively an induced insertion function f_I^{str} from f_I: $f_I^{str}(\epsilon) = \epsilon$ and $f_I^{str}(s_n) = f_I(\epsilon, e_1) f_I(e_1, e_2) \cdots f_I(e_1 e_2 e_{n-1}, e_n)$ where $s_n = e_1 e_2 \cdots e_n \in E_o^*$.

The modified output L_{out} of the system G under the impact of the insertion function f_I is then given by

$$\begin{aligned} L_{out} :&= f_I^{str}(P[L(G)]) \\ &= \{\tilde{s} \in (E_i^* E_o)^* | \exists s \in P[L(G)] : \tilde{s} = f_I^{str}(s)\}. \end{aligned} \tag{3}$$

To pursue succinct notations, we use f_I and f_I^{str} interchangeably in the sequel. Specifically, in this paper we are looking for the insertion functions that satisfy the private enforceability [26].

Definition 2 (Private Enforceability): Given a DES G and the observation mask P, an insertion function f_I is privately enforcing if (i) admissibility: $\forall se_o \in P[L(G)], s \in E_o^*, e_o \in E_o$, $\exists s_I \in E_I^*$ such that $f_I(s, e_o) = s_I e_o$; (ii) private safety: $L_{out} \subseteq L_{safe} = P[L(G)] \backslash (P[L(G)] \backslash P(L_{NS})) E_o^*$, where $L_{NS} = \{t \in L(G, X_0) | \exists x_i \in X_0, f(x_i, t) \cap X_{NS} \neq \emptyset\}$.

Intuitively, the admissibility requires that the insertion function f_I is well defined on all the strings from $P[L(G)]$. The private safety requirement restricts the modified output L_{out} from f_I to the non-secret behavior of the system L_{safe}, which is the set of projected strings that never reveal the secret. Therefore, by the definition, a private enforcing insertion function guarantees CSO.

III. Enforcement of Decentralized Opacity via Insertion Functions

In this section, we focus our study on opacity problems of DES with a decentralized architecture. Specifically, we extend the investigation of opacity-enforcement strategies to the case in which the system G can be observed by multiple intruders as shown in Fig. 2.

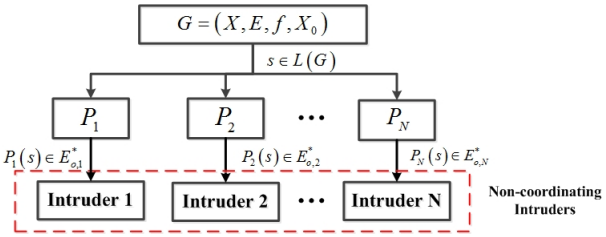


Fig. 2: The DES G observed by non-coordinating intruders.

We first consider the case where no coordination exists among intruders. Let I_i, $i \in \mathcal{N} = \{1, 2, \ldots, N\}$ denote a team of N intruders. Similar to the centralized scenario, each intruder has a complete prior knowledge of the system G. Intruder I_i is associated with the locally observable events $E_{o,i} \subseteq E$, $i \in \mathcal{N}$. The partial observation for I_i is characterized by the projection $P_i : E^* \to E_{o,i}^*$ when no insertion function exists. The property of *decentralized current-state opacity* is formally defined as follows.

Definition 3 (Decentralized CSO (D-CSO)): Given the set of observable events $E_{o,i} \subseteq E$ for intruder I_i, $i \in \mathcal{N}$, the secret state set X_S, and the non-secret state set X_{NS}, the system $G = (X, E, f, X_0)$ is said to be decentralized current-state opaque with respect to $E_{o,i}$ $i \in \mathcal{N}$, X_S and X_{NS} if

$$\begin{aligned} &(\forall i \in \mathcal{N})(\forall x_{0,i} \in X_0)(\forall t_i \in L(G, x_{0,i}) : f(x_{0,i}, t_i) \in X_S) \\ &\Rightarrow (\exists x'_{0,i} \in X_0)(\exists t'_i \in L(G, x'_{0,i}))[(f(x'_{0,i}, t'_i) \in X_{NS}) \wedge \\ &(P_i(t'_i) = P_i(t_i))]. \end{aligned} \tag{4}$$

The D-CSO of G suggests that any secret state in X_S be not inferred by any one of the intruders. It follows from Definition 3 that D-CSO can be viewed as a decentralized counterpart of CSO, which implies that enforcing D-CSO for G with multiple intruders is equivalent to enforcing CSO with respect to each individual intruder I_i, $i \in \mathcal{N}$. Motivated by this fact, we can synthesize local opacity-enforcing insertion function f^i for I_i, $i \in \mathcal{N}$ independently.

We illustrate the idea of synthesizing appropriate insertion functions for each intruder by the following example.

Example 1: Consider $G = (X, E, f, X_0)$ shown in Fig. 3, where $E = \{a, b, c, d\}$. The secret states are $X_S = \{1, 2, 6\}$,

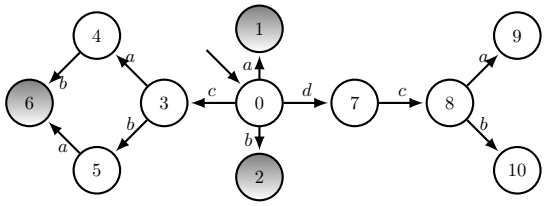
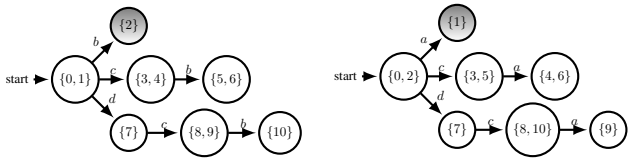


Fig. 3: An illustrative example, the secret states are shaded



(a) obs_1 (b) obs_2

Fig. 4: Observers for the system G.

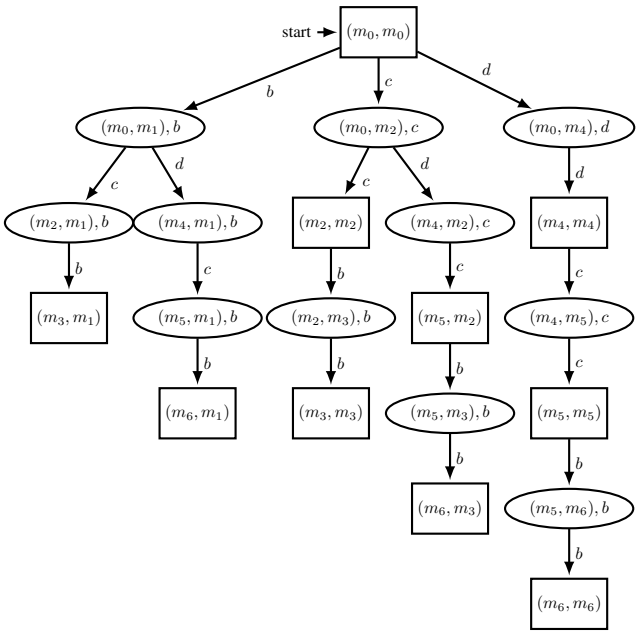
which are the shaded states in Fig. 3. We assume that G is observed by two intruders with different observation projections induced by $E_{o,1} = \{b, c, d\}$ and $E_{o,2} = \{a, c, d\}$, respectively. The observer obs_1 and obs_2 can be constructed in a standard way [9] as shown in Fig. 4. Each state in obs_i contains the current state estimation of intruder I_i, $i = 1, 2$. From Fig. 4, both observers reveal some secrets (the shaded states in Fig. 4) without the opacity-enforcement mechanism.

Since there is no coordination between the intruders, we follow the procedures in [27] to construct the all insertion structure (AIS) that encodes all the valid system and insertion function moves for each intruder respectively. It is then possible to extract an insertion function from the AIS.

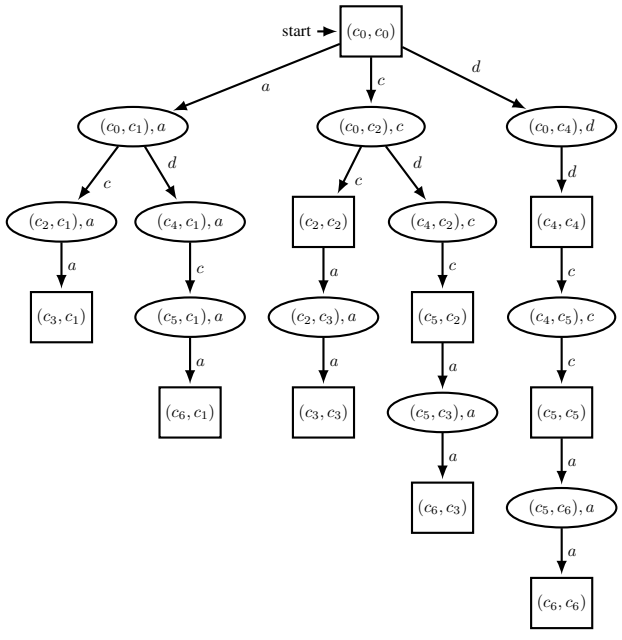
The AIS $(Q, E_o \cup \{\epsilon\}, f, q_0)$ can be seen as a game structure between the system and the insertion function, where $Q = Q_S \cup Q_I$, Q_S denotes the system state set and Q_I denotes the insertion function state set. Each $q \in Q_S$ has a pair of state estimates, the first one is the intruder's estimate, which could be wrong due to the inserted events, and the second estimate is the real system estimate. For each $q \in Q_I$, besides the intruder and system's state estimate, it also consists of current system output from G. $f(q, e) = q'$ for $q, q' \in Q$ and $e \in E_o \cup \{\epsilon\}$ represents the transition function, $q_0 \in Q_S$ is the initial state. As shown in Fig. 5, the rectangles represent the system states and the ellipses represent the insertion function states. All the transitions with events originated from the system states are system moves that are not controllable, while all the transitions with events from insertion function states are insertion function moves that the intruder actually observes.

Theorem 1: [22] CSO is privately enforceable if and only if the AIS is nonempty.

Remark 3: The main differences of our paper's AIS definition from [27] are two folds. The first is that we unfold the moves of the insertion function as well as the intruder's state estimate *event by event*, while in [27], the insertion function's move is from $E_o^* \cup \{\epsilon\}$, which could denote the whole string that has been inserted. For example, in our AIS definition, if we have a transition $q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{c} q_3$, where $q_0, q_1, q_2 \in Q_I$ and $q_3 \in Q_S$, in [27]'s definition, the same transition would be simplified to $q_0 \xrightarrow{abc} q_3$. The second is that, if the system G contains loops (for the simplest case, imagine there is a self-loop in some state), it could be the case that the inserted string contains s^* for some $s \in E_o^*$ and becomes arbitrarily long (for example, Fig. 7 in [27]). In our paper, we restrict the inserted strings to be $*$-free, that is, the insertions cannot be arbitrarily long and we replace s^* with ϵ. Our definition with unfolding and $*$-free in insertions are to facilitate the analysis of joint opacity enforcement in Section IV.



(a) AIS_1



(b) AIS_2

Fig. 5: AISs for intruders I_1 and I_2.

The AISs for intruders I_1 and I_2 in our motivating example

are as shown in Fig 5, where $m_0 = \{0,1\}, m_1 = \{2\}, m_2 = \{3,4\}, m_3 = \{5,6\}, m_4 = \{7\}, m_5 = \{8,9\}, m_6 = \{10\}$, $c_0 = \{0,2\}, c_1 = \{1\}, c_2 = \{3,5\}, c_3 = \{4,6\}, c_4 = \{7\}, c_5 = \{8,10\}, c_6 = \{9\}$. For instance, in AIS_1 shown in Fig. 5 (a), starting from the initial state where the intruder and the system's estimates are (m_0, m_0). If the event b occurs in the system, AIS_1 transits to the insertion function state $((m_0, m_1), a)$ since the system observer sees the event b and the intruder observer observes nothing as the insertion has not been decided yet. Then if the insertion function decides to insert c, the system transits to the insertion function state $((m_2, m_1), c)$ as the intruder observer observes c and the system observer will ignore the insertion function outputs. Then the real system output b is appended and consequently the AIS transits to the system state (m_3, m_1).

Theorem 2: Given the system G and N intruders with observation mask P_i, $i \in \mathcal{N}$, D-CSO is privately enforceable if and only if AIS_i is nonempty for all $i \in \mathcal{N}$.

Proof: On the one hand, D-CSO holds if and only if local CSO holds for all $i \in \mathcal{N}$. On the other hand, for each intruder I_i, local CSO is privately enforceable if and only if the AIS_i is not empty by Theorem 1. Therefore the proof is completed. ■

IV. Synthesis of Insertion Functions for Joint Opacity Enforcement

Rather than observing the same system without coordination, in many applications, intruders do coordinate among themselves by exchanging their estimates of the system's states. For these applications, decentralized opacity notions of coordinated intruders need further investigation.

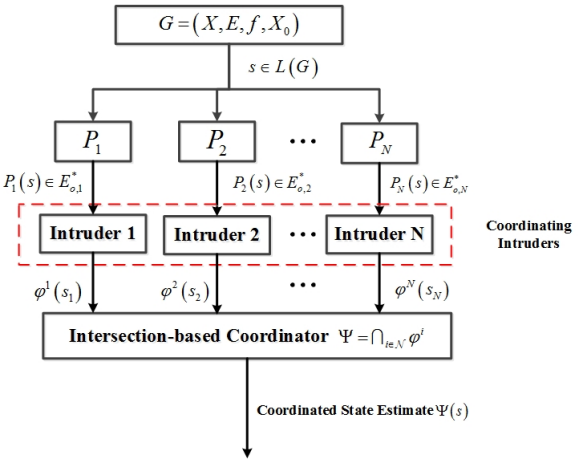


Fig. 6: The DES G observed by intruders with intersection-based coordination protocols.

A. Intersection-based Coordination Protocol

In this section, we investigate intruders that may coordinate with each other via an *intersection-based* protocol [25]. As shown in Fig. 6, we assume that the team of intruders I_i, $i \in \mathcal{N}$ not only generate local state estimates but report the estimates to a *coordinator* as well. The coordinator has no knowledge about the system. It forms the so-called *coordinated estimate* by taking the intersection of the local estimates it receives. The communication from the local intruders to the coordinator is assumed to have no delay. The collaboration is restricted by the following rules: (1) intruders have no knowledge of the projections of one another; (2) the only collaboration between the intruders is through the coordinator.

Before proceeding to opacity issues in the coordinated decentralized framework, we first study the intersection-based coordination protocol in Fig. 6. For the intruder I_i, $i \in \mathcal{N}$, a string-based local estimation map $\psi^i : P_i[L(G)] \to 2^X$ is defined as follows: for $s \in L(G)$ and $s_i := P_i(s)$, $\psi^i(s_i) = f\left(x_0, P_i^{-1}(s_i) \cap L(G)\right)$.

Then, we define an *intersection-based* coordination protocol $\Psi : \Pi_{i \in \mathcal{N}} P_i[L(G)] \to 2^X$ as

$$\Psi(s_1, s_2, \cdots, s_N) = \bigcap_{i \in \mathcal{N}} \psi^i(s_i) \tag{5}$$

Intuitively, the coordination protocol Ψ takes the intersection of the local estimates reported by the intruders and forms a *coordinated estimate* accordingly.

B. Enforcement Scheme of Joint Opacity in Discrete-event Systems

We now consider opacity issues of DES that can be observed by intruders following the intersection-based coordination protocol in Eq. 5. Roughly speaking, the system is said to be *jointly current-state opaque* if no coordinated estimate ever reveals the secret information.

Definition 4 (Joint CSO (J-CSO)): Given the set of observable events $E_{o,i} \subseteq E$ for intruder I_i, $i \in \mathcal{N}$, the secret state set X_S, the non-secret state set X_{NS} and the intersection-based coordination protocol Ψ, the system $G = (X, E, f, X_0)$ is said to be jointly current-state opaque with respect to $E_{o,i}$, $i \in \mathcal{N}$, X_S and X_{NS} if for each intruder, local CSO holds and

$$\Psi(s_1, s_2, \cdots, s_N) \cap X_S \neq \emptyset \wedge \Psi(s_1, s_2, \cdots, s_N) \cap X_{NS} \neq \emptyset \tag{6}$$

In this paper, we present a centralized approach to synthesize the individual insertion functions to enforce J-CSO. The following example shows that, in general, local insertion functions that enforce D-CSO of a system may not enforce J-CSO.

Example 2: With AIS_1 and AIS_2 in Fig. 5, D-CSO is guaranteed in Example 1. However, if the two intruders can send their estimates to the intersection-based coordinator, joint opacity may be violated. For instance, from Fig. 5, if the string cab happens in the system, it will be projected to be cb and ca for intruders 1 and 2, respectively. If both insertion functions choose not to insert anything, which are valid moves from their local AISs, the resulting estimates reported by intruders 1 and 2, after observing cb and ca, are $\{5,6\}$ and $\{4,6\}$, respectively. As our coordinator performs the intersection of the estimation, it will result in $\{6\} \in X_S$, which reveals a secret.

Example 2 implies that insertion functions that enforce D-CSO do not necessarily guarantee J-CSO. Therefore, the

insertion functions need to be specifically coordinated to enforce the J-CSO.

Our first step is to encode the AIS into a corresponding Nondeterministic Finite-state Mealy machine (NFM) for a concise representation.

Definition 5: An NFM is a 5-tuple

$$\mathcal{M} = (Q, \Sigma_{In}, \Sigma_{Out}, q_0, f_{NMF}), \tag{7}$$

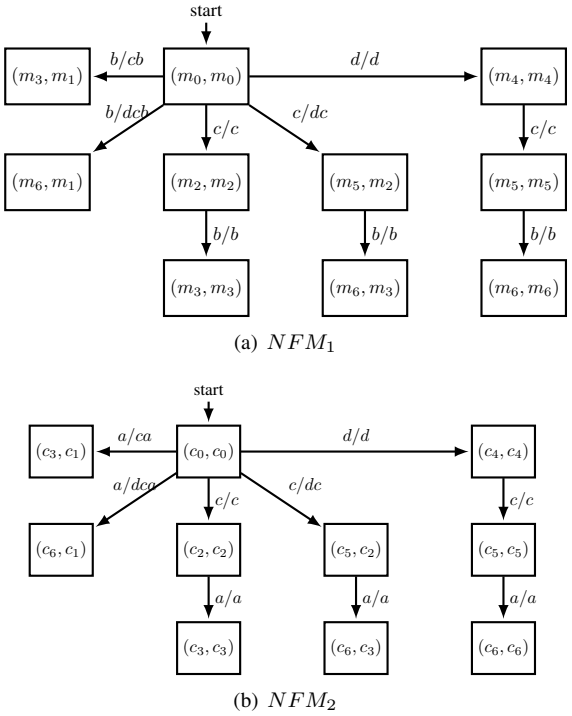
where Q is the set of states, Σ_{In} and Σ_{Out} are the sets of input and output symbols, respectively, $q_0 \in Q$ is the initial state, $f_{NMF}(q, e) = (q', o)$ defines the transition and input output relation for $q, q' \in Q, e \in \Sigma_{In}, o \in \Sigma_{Out}$.

The nondeterminism of an NFM comes from the fact that in general $|f_{NMF}(q, e)| \geq 1$, which implies that the same input on the same state may result in non-unique insertions and transit to different states. Our NFM formulation is similar to the insertion automaton [27] but we allow nondeterministic choices of insertions upon observing a system output $e \in E_o$. The procedure to convert an AIS into an NFM $\mathcal{M} = (Q, \Sigma_{In}, \Sigma_{Out}, q_0, f)$ is as follows. Q is the set of all the systems states of AIS. Σ_{In} is the set of all the events from system states and Σ_{Out} is the set of all the possible insertion strings. The transition function is defined as $f(q, e) = (q', o)$, where $o = o' + e, o' \in E_I^*, e \in E_o$, o' is the inserted string, e is the system input and o denotes the output from the state q when the system input is e.

Example 3: Fig. 7 denotes the NFMs corresponding to AIS_1 and AIS_2 in Fig. 5, respectively. Note that, different from AIS, in the NFM formulation, upon observing an event e, the state directly jumps from q to q' while outputting the string o. However, what really happens, as shown in AIS, is that the intruder's estimation is updated event by event for each output of the insertion function. Such estimation evolution is omitted in the NFM formulation for conciseness but can be recovered from our AIS.

To keep the NFMs synchronized with the original system that intruders try to compromise, we construct another system observer obs as a DFA with $E_{obs,o} = \bigcup_{i \in \mathcal{N}} E_{o,i}$. That is, if an event is observable to any one of the intruders, it is observable to this system observer. In our example, $E_{obs,o} = E$ and the observer has the identical structure with the original system as shown in Fig. 3. The observer obs can be viewed as an NFM that outputs empty string ϵ for all inputs. Given N AISs' in the form of NFM $\mathcal{M}_i = (Q_i, \Sigma_{In}^i, \Sigma_{Out}^i, q_0^i, f_i)$ for $i \in \mathcal{N}$ and the system observer obs, we can obtain the composed NFM $\mathcal{G} = (Q_G, \Sigma_{In}, \Sigma_{Out}, q_0, f)$ that describes all the possible combined insertion behaviors, where $Q = Q_1 \times Q_2 ... \times Q_N \times Q_{obs}$, $\Sigma_{In} = \Sigma_{obs,In} = E_{obs,o}$, $\Sigma_{Out} = \Sigma_{1,Out} \times \Sigma_{2,Out} ... \times \Sigma_{N,Out} \times \{\epsilon\}$. The transition relation $f((q_1, q_2, ..., q_N, q_{obs}), e) = ((q_1', q_2', ..., q_N', q_{obs}'), o)$ is given by

- $o = (o_1, ..., o_N, \epsilon)$
- $(q_{obs}', \epsilon) = f_{obs}(q_{obs}, e)$
- $(q_i', o_i) = f_i(q_i, e)$ if $e \in E_{o,i}$; otherwise $(q_i', o_i) = (q_i, \epsilon)$



(a) NFM_1

(b) NFM_2

Fig. 7: The NFMs converted from AISs.

While constructing this product NFM $\mathcal{G}$, we assume that when an event $e \in \Sigma_o$ in the system occurs, it is guaranteed that for every intruder I_i such that $e \in E_{o,i}$ holds, its corresponding insertion function will finish outputting the modified string o_i before the next system event $e' \in E_o$ is generated. It is always possible since we restrict the output of the insertion functions to be $*$-free. In this regard, every insertion function is synchronized with the system inputs.

Example 4: Fig. 8 illustrates the $\mathcal{G}$ from Example 2 and the corresponding NFM_1 and NFM_2 in Fig. 7. For simplicity, in this figure we omit the constant output ϵ from the system observer as well as each individual observer's state estimation.

For a given transition $f((q_0, q_1, ..., q_{n-1}, q_{obs}), e) = ((q_0', q_1', ..., q_{n-1}', q_{obs}'), o)$, it is then possible to check whether the secrets will be revealed and joint opacity could be violated during this transition with the help of the AISs, since as mentioned earlier, the event by event evolution of the state estimation for each intruder upon observing a modified string is omitted in the NFM but not AIS.

For example, in the NFM from Fig. 8, starting from the initial state, when the event c happens and the insertion functions decide to insert d and ϵ respectively, the transition is $(m_0, c_0, 0) \xrightarrow{c/(dc,c)} (m_5, c_2, 3)$. From the AISs in Fig. 5, the evolution of each intruder's estimation can be see as a two step transition $(m_0, c_0, 0) \xrightarrow{d,c} (m_4, c_2, 3) \xrightarrow{c,\epsilon} (m_5, c_2, 3)$. Note that there is an intermediate state $(m_4, c_2, 3)$ that is not shown in $\mathcal{G}$. In the first step, upon observing the system event c, the first insertion function outputs d and the second insertion function, since it decides to insert nothing, the system event c is directly outputted. Therefore, the

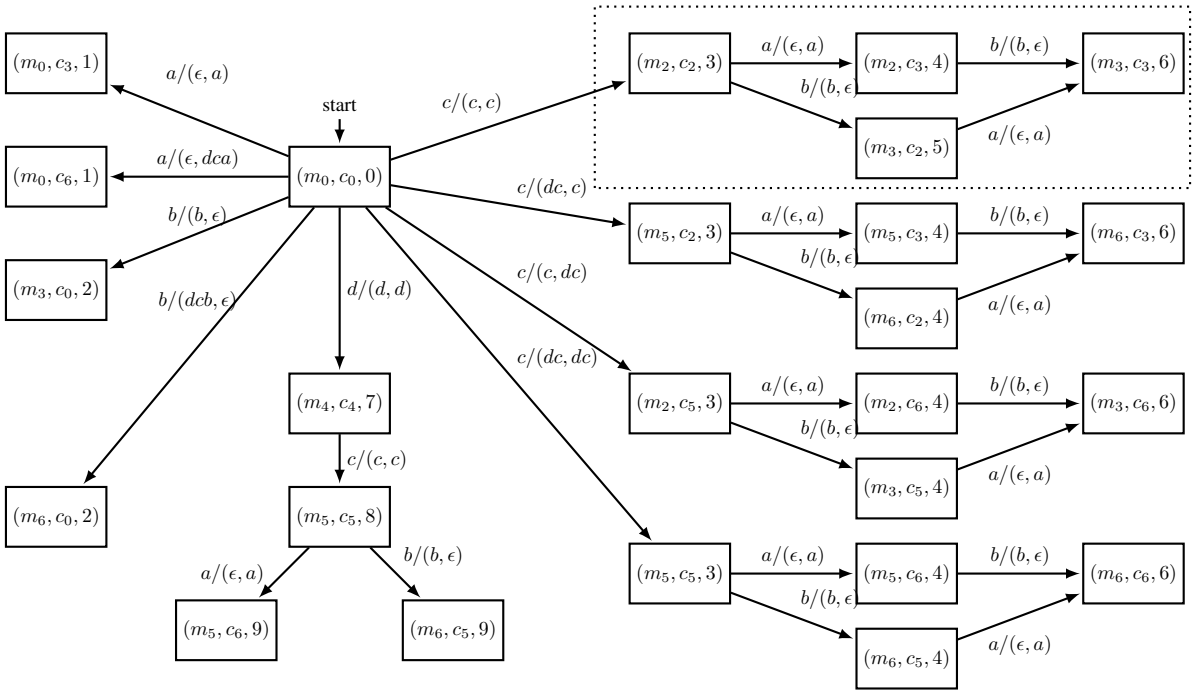


Fig. 8: The NFM $\mathcal{G}$, the states in the dashed box are pruned

estimations evolve from m_0 to m_4, c_0 to c_2, and the system observer's estimation changes from 0 to 3. In the second step, the first insertion function outputs the system event c and the second insertion function outputs ϵ. It can be seen that our assumption is that the event output (including ϵ) for each intruder is synchronized.

To determine if a transition $(q, e) \to (q', o)$ in $\mathcal{G}$ is safe, we examine every intermediate state from the AISs that evolves with each output event to see if the joint estimation reveals a secret. By definition, q_{obs} encodes the *true* set of states that the system is currently in. The first element of AIS_i state — we denote as est_i — represents each *intruder's estimation* of the current state. According to our coordination rule, the joint estimation J_{est} is then obtained by taking intersection among q_{obs} and $est_i, i = 1, \ldots, N$.

$$J_{est} = q_{obs} \cap est_0 \cap ... \cap est_{n-1}$$

Proposition 1: J_{est} does not reveal a secret if $J_{est} \cap X_S \neq \emptyset \implies J_{est} \cap X_{NS} \neq \emptyset$

We now define J-CSO in the presence of synthesized insertion functions as follows.

Definition 6: Given N intruders with unobservable event sets $E_{uo,1}, E_{uo,2}, ..., E_{uo,N}$ and their insertion functions $f_{I,1}, f_{I,2}, ..., f_{I,N}$, the system is J-CSO against the intruders if

- For each individual intruder i, the insertion function $f_{I,i}$ enforces local CSO.
- The J_{est} never reveals the secret.

Furthermore, we define J-CSO to be jointly privately enforceable if all individual insertion functions are locally privately enforcing and J_{est} never reveals the secret.

An intermediate state is unsafe if its J_{est} reveals a secret. A transition $(q, e) \to (q', o)$ in $\mathcal{G}$ is unsafe if any of the intermediate states between q and q' is unsafe. Similarly, any state $q \in Q$ of $\mathcal{G}$ is unsafe if its J_{est} reveals a secret. If a transition is found to be unsafe, it will be pruned. If a state $q \in Q$ is found to be unsafe, this state, together with all its incoming and outgoing transitions, will be pruned. If after the pruning, at some state $q' \in Q$, there is no incoming transition (except the initial state) or there is no outgoing transition defined on an event e that could happen in this state, which implies that the system blocks when e happens at q' since there is no insertion function available, then such state is also unsafe and all its incoming and outgoing transitions will be pruned. Again, such pruning may trigger new deadlocks and create unsafe states. Therefore, this is an iterative process until no unsafe state is found or the initial state is pruned.

For example, as shown in Fig. 8, the state $(m_3, c_3, 6)$ is an unsafe state that reveals the secret. Because $m_3 \cap c_3 \cap \{6\} = \{5, 6\} \cap \{4, 6\} \cap \{6\} = \{6\} \in X_S$. Therefore it has to be pruned, which results in the states $(m_2, c_3, 4)$ and $(m_3, c_2, 5)$ being unsafe since there are no outgoing transitions any more. Consequently, pruning $(m_2, c_3, 4)$ and $(m_3, c_2, 5)$ makes $(m_2, c_2, 3)$ unsafe. After deleting $(m_2, c_2, 3)$, the pruning process stops. Since no more state or transitions is found to be unsafe, the resulting $\mathcal{G}$ can be found in Fig. 8, excluding the states in the dashed box.

Theorem 3: Given the system model G and N intruders with observation projections P_i, $i \in \mathcal{N}$, J-CSO is jointly privately enforceable if and only if $\mathcal{G}$ is nonempty after pruning.

Proof: If J-CSO is jointly privately enforceable, in our definition, it implies that for each individual intruder I_i, local opacity is privately enforceable and thus AIS_i is nonempty by Theorem 1. Since AIS_i encodes all the possible local insertion functions that are privately enforcing, $\mathcal{G}$ as

the product of AISs encodes all the possible joint insertion strategies that are privately enforcing. Since J-CSO is jointly privately enforceable, there exists at least one local insertion function for each intruder I_i that is privately enforcing and the joint estimate never reveals the secret. Thus the joint insertion strategy is nonempty, which implies that $\mathcal{G}$ is nonempty.

Conversely, non-emptiness of $\mathcal{G}$ implies that AIS_i is nonempty for any i. Thus, the local opacity is guaranteed. Furthermore, since $\mathcal{G}$, after pruning, encodes all the valid privately enforcing insertion functions for each intruder such that the joint state estimate never reveals the secret, J-CSO is guaranteed. ■

C. Complexity Analysis

Given N intruders and the system G with $|X|$ states, the space and time complexity to construct each AIS is polynomial with $|X_{\mathcal{E}}|$ [27], where $|X_{\mathcal{E}}| = 2^{|X|}$ denotes the total number of states of the state estimator. Each NFM's state space is at most the state pace of its AIS. Therefore, the space complexity to construct $\mathcal{G}$ is polynomial in $|X_{\mathcal{E}}|$ and exponential in $|\mathcal{N}|$. The pruning process, in the worst case, looks over all the states in $\mathcal{G}$ and intermediate states, which is also polynomial in $|X_{\mathcal{E}}|$ and exponential in $|\mathcal{N}|$. So to sum up, the space and time complexity in our proposed centralized synthesis approach are both polynomial with $|X_{\mathcal{E}}|$ and exponential in $|\mathcal{N}|$.

V. Conclusion

In this paper, we investigate the opacity-enforcement problem for discrete-event systems that can be observed by multiple intruders. The major contribution of this paper is summarized as follows. First, we introduce opacity notions for two cases of DES in the presence of multiple intruders, one with coordination and the other without coordination. Next, we adopt the event insertion mechanism to ensure decentralized opacity for intruders without coordination; the synthesized insertion functions are further refined to enforce joint opacity of DES when intruders can coordinate via an intersection-based protocol. Future research directions may include: (i) introducing notions of joint opacity corresponding to other types of coordination protocols among the intruders; (ii) development of algorithms for enforcing other notions of joint opacity with respect to the new types of coordination protocols.

References

[1] M. A. Bishop, *Computer Security: Art and Science*. Boston: Addison-Wesley, 2003.

[2] L. Mazaré, "Using unification for opacity properties," *Proceedings of the 4th IFIP WG1*, vol. 7, pp. 165–176, 2004.

[3] S. Kumari and M. K. Khan, "More secure smart card-based remote user password authentication scheme with user anonymity," *Secur. Commun. Netw.*, vol. 7, no. 11, pp. 2039–2053, 2014.

[4] N. B. Hadj-Alouane, S. Lafrance, F. Lin, J. Mullins, and M. M. Yeddes, "On the verification of intransitive noninterference in mulitlevel security," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 5, pp. 948–958, 2005.

[5] R. Alur, P. Černỳ, and S. Zdancewic, "Preserving secrecy under refinement," *Automata, Languages and Programming*, pp. 107–118, 2006.

[6] A. Rabbachin, A. Conti, and M. Z. Win, "Wireless network intrinsic secrecy," *IEEE/ACM Tran. Netw.*, vol. 23, no. 1, pp. 56–69, 2015.

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

[8] 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.

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

[10] J. W. Bryans, M. Koutny, and P. Y. A. Ryan, "Modelling opacity using petri nets," *Electronic Notes in Theoretical Computer Science*, vol. 121, pp. 101–115, 2005.

[11] 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, 2017.

[12] A. Saboori and C. N. Hadjicostis, "Notions of security and opacity in discrete event systems," in *Proc. 46th IEEE Conf. Decision and Control*. IEEE, 2007, pp. 5056–5061.

[13] ——, "Current-state opacity formulations in probabilistic finite automata," *IEEE Trans. Autom. Control*, vol. 59, no. 1, pp. 120–133, 2014.

[14] ——, "Verification of k-step opacity and analysis of its complexity," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 3, pp. 549–559, 2011.

[15] ——, "Verification of infinite-step opacity and complexity considerations," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1265–1269, 2012.

[16] ——, "Opacity-enforcing supervisory strategies via state estimator constructions," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1155–1165, 2012.

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

[18] 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, 2016.

[19] M. Ben-Kalefa and F. Lin, "Opaque superlanguages and sublanguages in discrete event systems," *Cybernetics and Systems*, vol. 47, no. 5, pp. 392–426, 2016.

[20] Y. Falcone and H. Marchand, "Enforcement and validation (at runtime) of various notions of opacity," *Discrete Event Dynam. Syst.: Theory Applicat.*, vol. 25, no. 4, pp. 531–570, 2015.

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

[22] 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.

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

[24] Y.-C. Wu and S. Lafortune, "Comparative analysis of related notions of opacity in centralized and coordinated architectures," *Discrete Event Dynam. Syst.: Theory Applicat.*, vol. 23, no. 3, pp. 307–339, 2013.

[25] A. Paoli and F. Lin, "Decentralized opacity of discrete event systems," in *Proc. 2012 American Control Conference (ACC)*. IEEE, 2012, pp. 6083–6088.

[26] Y.-C. Wu, G. Lederman, and S. Lafortune, "Enhancing opacity of stochastic discrete event systems using insertion functions," in *Proc. 2016 American Control Conference (ACC)*. IEEE, 2016, pp. 2053–2060.

[27] Y.-C. Wu and S. Lafortune, "Synthesis of optimal insertion functions for opacity enforcement," *IEEE Trans. Autom. Control*, vol. 61, no. 3, pp. 571–584, 2016.