2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Maximum Information Release While Ensuring Opacity in Discrete Event Systems

Bo Zhang[1], Shaolong Shu[1], and Feng Lin[1,2]

Abstract—**Opacity is important in investigating secrecy, privacy, and other important properties in general systems that can be modeled as discrete event systems. To ensure opacity, a controller may be used to control information released to the public. For transparency and other reasons, it is often desired the information released to the public be maximum, as long as opacity is not violated. In this paper, we investigate how to release the maximum information while ensuring opacity. We find a necessary and sufficient condition for a control policy to ensure opacity. We also develop methods and algorithms to design a controller that releases maximum information. We consider both strong opacity and weak opacity.** [1]

I. Introduction

Opacity is an important property to be investigated in many practical systems such as computer systems, communication networks, and databases systems in which information flow must be controlled to ensure secrecy, security, privacy, anonymity, non-interference, etc. Because of its importance, opacity has been studied in the framework of discrete event systems. In particular, [3] uses the model of labeled transition systems to study opacity. A generalized definition of opacity is proposed. It establishes links between opacity and the information flow concepts of anonymity and non-inference. In [11], two notions of opacity are investigated. A system is (S, P)-opaque if the evolution of the system through a set of secret states S remains opaque to an observer who observes activity in the system through the projection P. A system is (S, P ,K)-opacity if (S, P)-opacity remains true for K observations following the departure of the system's state from the set S.

In [12] [13], the problems of checking opacity and synthesizing opaque systems by selecting the set of observable events are addressed. It shows that checking whether a system is opaque and computing an optimal static observer ensuring opacity are both $PSPACE$-complete problems.

To ensure opacity, [14] investigates the problem of constructing a minimally restrictive opacity-enforcing supervisory, which restricts system within some legal behaviors while enforcing initial-state opacity. In [7], [8], the problem of computing a controller that enforces the opacity of a predicate against an attacker is addressed. It shows that an optimal control always exists. It also provides sufficient conditions under which the solution is regular and can be effectively computed.

We investigate opacity of discrete event systems in [9]. Unlike the previous works, we define opacity of a language with respect to another language. This two-language approach give us more flexibility in applying opacity in practical problem. Using our definition in [9], opacity can not only be used to study security and privacy problems, but also other information flow problems such as observability [10], diagnosability [15], and detectability [16]. While most previous works define opacity in its strong sense, we define both strong opacity and weak opacity [9]. We say that a language L is strongly opaque with respect to another language K if all strings in L are confused with some strings in K from an external agent's point of view. We say that a language L is weakly opaque with respect to another language K if some strings in L are confused with some strings in K. The negation of weak opacity is called no opacity (that is, totally transparent). Strong opacity, weak opacity and no opacity can be used in different applications to solve different problems. For example, the dining cryptographers problem [6] involves both strong opacity and weak opacity. Checking weak opacity (and hence no opacity) can be done in polynomial complexity [18]. We further investigate opaque superlanguages and sublanguages in [1] and supervisory control to ensure opacity in [2].

In this paper, we investigate the problem of maximum information release while ensuring opacity (weak or strong) of discrete event systems. The rational for the investigation is as follows. When an authority wants to decide what information can be released to the public, two objectives are often considered: (1) For security reasons, certain things must be kept secret. This means either strong opacity or weak opacity must be ensured. (2) For transparency reasons, the more information is released to the public, the better. For example, in USA, the Freedom of Information Act (FOIA) requires that certain information and records of government agencies to be released to the public upon request, unless such release will harm national security or covered under other nine specific exemptions. Since national security can be modeled as an opacity problem, what information can be released can be determined by solving a maximum information release problem.

The Maximum Information Release Problem can be formulated as follows. An external agent is trying to get as much information as possible in order to distinguish strings in L and K. What information will be released or communicated

[1]School of Electronics and Information Engineering, Tongji University, Shanghai, China. [2]Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202, USA, E-mail: flin@ece.eng.wayne.edu. The authors of this paper are supported in part by the National Natural Science Foundation of China under Grants 60904019, 61143006 and 71071116.

978-1-4799-3684-7/14/$31.00 ©2014 IEEE

to the external agent is controlled by an internal controller (or internal agent). The objective of the internal controller is to ensure that opacity (either strong opacity or weak opacity) is preserved. Under this condition, the internal controller is also required to release or communicate as much information as possible for the sake of transparency, fairness or other purposes. In the framework of discrete event systems, the maximum information release problem is for the internal controller to communicate as many occurrences of observable transitions to the external agent while ensure either strong opacity or weak opacity.

The Maximum Information Release Problem can be solved either on-line or off-line. In this paper, we focus on off-line approach. We derive a necessary and sufficient condition for an information release policy to ensure strong opacity. We then propose algorithm to synthesize an information release policy for strong opacity off-line. The computational complexity of the algorithm is also analyzed. We then do the same for weak opacity.

Our approach is new and innovative. It is different from the approaches existed in the literature. The results that are most closely related to ours are those published in [4], [5]. In [4], [5], opacity is defined for a secret predicate φ. A system is opaque if every run in φ is observationally equivalent to a run not in φ. This definition is a special case of our definition of strong opacity. To ensure opacity, a static or dynamic mask is used to hide the occurrence of (observable) events. Various problems and their solutions are then discussed in [4], [5]. The main differences between [4], [5] and this paper are as follows. (1) The definitions of opacity are different. The opacity defined in [4], [5] (and most other papers on opacity) is a special case of our strong opacity. Weak opacity is not discussed in [4], [5]. (2) The mechanisms of control are different. The dynamic mask in [4], [5] requires that if an occurrence of an event is hidden, then it cannot change the state of the dynamic mask. We do not need this assumption. This makes our controller more flexible and realistic. (3) We develop algorithms to solve the Maximum Information Release Problem for both strong opacity and weak opacity. Another approach is presented in [17]. Instead of "deleting" occurrences of events as in [4], [5] and this paper, [17] uses insertion function to ensure opacity. This is a very interesting approach and can be viewed as "dual" to deletion approach.

Due to space limitation, all the proofs are omitted. They can be obtained from the authors.

II. Opacity of Discrete Event Systems

In this paper, we model a discrete event system by a *deterministic automaton* [9]

$$G = (Q, \Sigma, \delta, q_0),$$

where Q is the set of state, Σ the set of events, δ the transition function, and $q_0 \in Q$ the initial state. The (partial) transition function $\delta : Q \times \Sigma \rightarrow Q$ describes the system dynamics: given states $q, q' \in Q$ and event $\sigma \in \Sigma$, $\delta(q, \sigma) = q'$ if the execution of σ from state q takes the system to state q'. Note that $\delta(q, \sigma)$ is undefined whenever the event σ cannot be executed from the state q. The transition function is extended to $\delta : Q \times \Sigma^* \rightarrow Q$ in the usual way. We also use δ to denote the set of transitions: $\delta = \{(q, \sigma, \delta(q, \sigma)) : q \in Q \wedge \sigma \in \Sigma\}$. The behavior of discrete event system G is described by the language generated by G defined as

$$\mathcal{L}(G) = \{s \in \Sigma^* : \delta(q_0, s)!\},$$

where $\delta(q_0, s)!$ means $\delta(q_0, s)$ is defined. Each string in $\mathcal{L}(G)$ represents a possible execution or trajectory of the system. $\mathcal{L}(G)$ describes all possible strings that can be generated by the system. A subset of $\mathcal{L}(G)$ is a language that describes a particular behavior or property of the system. A language $L \subseteq \mathcal{L}(G)$ is (prefix) closed if any prefix of any string of the language also belongs to the langauge.

Opacity is defined with respect to two languages $K \subseteq \mathcal{L}(G)$ and $L \subseteq \mathcal{L}(G)$. Without loss of generality, we assume that K and L are marked by $Q_K \subseteq Q$ and $Q_L \subseteq Q$ respectively, that is,

$$K = \mathcal{L}_{m,K}(G) = \{s \in \mathcal{L}(G) : \delta(q_0, s) \in Q_K\}$$
$$L = \mathcal{L}_{m,L}(G) = \{s \in \mathcal{L}(G) : \delta(q_0, s) \in Q_L\}.$$

For convenience, we extend G by adding Q_K and Q_L as follows.

$$G = (Q, \Sigma, \delta, q_0, Q_K, Q_L).$$

To study opacity of discrete event systems, we consider a general observation mapping

$$\theta : \Sigma^* \rightarrow \Sigma^*.$$

θ is interpreted as follows: If a string of events s occurs in the system, an external agent or public will see $\theta(s)$. The natural projection $P : \Sigma^* \rightarrow \Sigma_o^*$ used in [10], where $\Sigma_o \subseteq \Sigma$ is a subset of observable events, is a special case of observation mapping. In general, however, θ can be any observation mapping, not restricted to the natural projection. An observation mapping θ can be extended from strings to languages as follows. For a language $L \subseteq \Sigma^*$, its mapping is defined as,

$$\theta(L) = \{t \in \Sigma^* : (\exists s \in L)\ t = \theta(s)\}.$$

For a language $J \subseteq \Sigma^*$, its inverse mapping is defined as

$$\theta^{-1}(J) = \{t \in \Sigma^* : \theta(t) \in J\}.$$

Two versions of opacities, strong opacity and weak opacity, are defined in [2] as follows. Given two languages $L, K \subseteq \mathcal{L}(G)$, L is strongly opaque with respect to K and θ, if

$$\theta(L) \subseteq \theta(K).$$

L is weakly opaque with respect to K and θ, if

$$\theta(L) \cap \theta(K) \neq \emptyset.$$

Since K and L are marked by $Q_K \subseteq Q$ and $Q_L \subseteq Q$ respectively, we can check strong opacity and weak opacity by calculating state estimates. After observing a string $t \in$

Σ^*, the state estimate, denoted by $SE^\theta(t)$, is the set of all possible states that the system may be in. Formally,

$$SE^\theta(t) = \{q \in Q : (\exists s \in \mathcal{L}(G))\theta(s) = t \wedge \delta(q_0, s) = q\}.$$

Proposition 1: L is strongly opaque with respect to K and θ if and only if

$$(\forall t \in \theta(\mathcal{L}(G)))SE^\theta(t) \cap Q_L \neq \emptyset \Rightarrow SE^\theta(t) \cap Q_K \neq \emptyset.$$

Proposition 2: L is weakly opaque with respect to K and θ if and only if

$$(\exists t \in \theta(\mathcal{L}(G)))SE^\theta(t) \cap Q_L \neq \emptyset \wedge SE^\theta(t) \cap Q_K \neq \emptyset.$$

From the above results, we conclude that the key to checking and ensuring strong or weak opacity is to calculate state estimates. If the observation mapping is the natural projection, then the state estimates can be obtained by constructing an observer as follows. Suppose that the discrete event system G is currently in a set of possible states $Q' \subseteq Q$, then the set of all possible states which the system may visit after observing $t \in \Sigma_o^*$ is denoted by

$$R(Q', t) = \{q \in Q : (\exists q' \in Q')(\exists s \in \Sigma^*)P(s) = t \\ \wedge \delta(q', s) = q\}.$$

In particular, the unobservable reach of Q' is defined as

$$UR(Q') = R(Q', \varepsilon).$$

The observer is a deterministic automaton

$$G_{obs} = (X, \Sigma_o, \xi, x_0) = Ac(2^Q, \Sigma_o, \xi, UR(\{q_0\})),$$

where $Ac(.)$ denotes the accessible part and the initial state $x_0 = UR(\{q_0\})$ is the unobservable reach of q_0. Note that a state $x \in X$ is a subset of Q ($x \subseteq Q$). The transition function $\xi : X \times \Sigma_o \rightarrow X$ is defined, for $x \subseteq Q$ and $\sigma \in \Sigma_o$, as:

$$\xi(x, \sigma) = UR(\{q \in Q : (\exists q' \in x)\delta(q', \sigma) = q\}).$$

If the above set is empty, then $\xi(x, \sigma)$ is undefined. We extend ξ to $\xi : X \times \Sigma_o^* \rightarrow X$ in the usual way. State estimates are characterized by the observer as stated in the following proposition.

Proposition 3: For the natural projection $\theta = P$, the state estimates are given by

$$SE^\theta(t) = \xi(x_0, t).$$

With these preparations, we investigate the maximum information release problem for opacity in the rest of the paper.

III. Maximum Information Release for Strong Opacity

Let us consider the following situation: The public (or an external agent) requests information of the system. What information can be release to the public is controlled by a controller. The situation is illustrated in Figure 1. The objective of the controller is to release as much information as possible to the public under the constraint that L is strongly opaque with respect to K. (We will consider weak opacity in Section V.) The controller can observe the set of observable events Σ_o. Its control is based on this observation. Formally, the control policy is described by a mapping

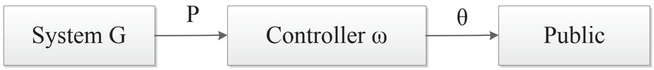

Fig. 1. Information release to the public

$$\omega : P(\mathcal{L}(G)) \rightarrow 2^{\Sigma_O}.$$

When the system G generates a string $s \in \mathcal{L}(G)$, the controller will see $P(s) \in P(\mathcal{L}(G))$. Its control decision $\omega(P(s)) \in \Sigma_o$ is the set of events whose occurrences will be released to the public (that is, the public knows when the event occurs). Hence, its external observation is described by a mapping

$$\theta^\omega : \mathcal{L}(G) \rightarrow \Sigma_o^*,$$

which is defined as follows.

$$\theta^\omega(\varepsilon) = \varepsilon, \qquad \theta^\omega(s\sigma) = \begin{cases} \theta^\omega(s)\sigma & \text{if } \sigma \in \omega(P(s)) \\ \theta^\omega(s) & \text{if } \sigma \notin \omega(P(s)) \end{cases}$$

Since the goal is to achieve maximum information release, let us define an order on ω as follows. Given two controls ω_1, ω_2, we say that $\omega_1 \leq \omega_2$ if

$$(\forall t \in P(\mathcal{L}(G)))\ \omega_1(t) \subseteq \omega_2(t).$$

In other words, ω_2 releases more information than ω_1. We say that $\omega_1 < \omega_2$ if

$$\omega_1 \leq \omega_2 \wedge (\exists t \in P(\mathcal{L}(G)))\ \omega_1(t) \subset \omega_2(t).$$

We now formally state the problem to be solved as follows. *Maximum Information Release Problem for Strong Opacity (MIRPSO)*

Find a control ω of information release that satisfies the following two conditions.

1) The observation mapping θ^ω corresponding to ω ensures strong opacity, that is,

$$\theta^\omega(L) \subseteq \theta^\omega(K).$$

2) For any other control ω' such that $\omega < \omega'$, the corresponding observation mapping $\theta^{\omega'}$ does not ensure strong opacity, that is,

$$\theta^{\omega'}(L) \not\subseteq \theta^{\omega'}(K).$$

To make the problem nontrivial, let us assume that if all information is released, then strong opacity is not satisfied. In other words,

$$P(L) \not\subseteq P(K).$$

Note that if all information is released, that is, for all $t \in P(\mathcal{L}(G))$, $\omega(t) = \Sigma_o$, then $\theta^\omega = P$.

To investigate the solution to MIRPSO, we note that, without further constraints, MIRPSO is difficult to solve, because there are simply too many choices for ω. For practical reasons, we assume that ω has a finite implementation (H, φ). Here H is a finite automaton

$$H = (Y, \Sigma_o, \eta, y_0),$$

where Y is the set of states, Σ_o the observable events, η the transition function, and $y_0 \in Y$ the initial state. H must satisfy the condition that $P(\mathcal{L}(G)) \subseteq \mathcal{L}(H)$. φ is a feedback mapping,

$$\varphi : Y \to 2^{\Sigma_o}.$$

The control $\omega : P(\mathcal{L}(G)) \to 2^{\Sigma_o}$ is obtained from H and φ as follows.

$$\omega(t) = \varphi(\eta(y_0, t))$$

Note that a sufficient condition for such a (H, φ) to exist is that ω is a right-congruence, that is,

$$(\forall t, t' \in \Sigma_o^*)\omega(t) = \omega(t') \Rightarrow (\forall v \in \Sigma_o^*)\omega(tv) = \omega(t'v).$$

If ω can be implemented by (H, φ), we denote it as $\omega = (H, \varphi)$. For $\omega = (H, \varphi)$, we can calculate state estimate $SE^{\theta^{\omega}}(t)$, which will be abbreviated as $SE^{\omega}(t)$ in the rest of the paper, as follows. Let

$$\tilde{G} = (\tilde{Q}, \Sigma, \tilde{\delta}, \tilde{q}_0) = G || H,$$

where $||$ denotes the parallel composition. Note that since $P(\mathcal{L}(G)) \subseteq \mathcal{L}(H)$,

$$\mathcal{L}(\tilde{G}) = \mathcal{L}(G) \cap P^{-1}\mathcal{L}(H) = \mathcal{L}(G)$$

A state of $\tilde{G}$ is a pair $\tilde{q} = (q, y)$. The set of events that the public or external agent can observe at state $\tilde{q}$ is given by $\varphi(y)$. Replace the transitions in $\tilde{G}$ that cannot be observed by public by ε as follows.

$$\tilde{G}_{\varepsilon} = (\tilde{Q}, \Sigma, \tilde{\delta}_{\varepsilon}, \tilde{q}_0),$$

where

$$\begin{aligned}\tilde{\delta}_{\varepsilon} = \{&(\tilde{q}, \sigma, \tilde{\delta}(\tilde{q}, \sigma)) : (\tilde{q}, \sigma, \tilde{\delta}(\tilde{q}, \sigma)) \in \tilde{\delta} \wedge \tilde{q} = (q, y) \\ &\wedge \sigma \in \varphi(y)\} \\ &\cup \{(\tilde{q}, \varepsilon, \tilde{\delta}(\tilde{q}, \sigma)) : (\tilde{q}, \sigma, \tilde{\delta}(\tilde{q}, \sigma)) \in \tilde{\delta} \wedge \tilde{q} = (q, y) \\ &\wedge \sigma \notin \varphi(y)\}\end{aligned}$$

$\tilde{G}_{\varepsilon}$ is a non-deterministic automaton with ε-transitions. We convert it to a deterministic automaton (observer) in the usual way:

$$\tilde{G}_{obs} = (\tilde{X}, \Sigma_o, \tilde{\xi}, \tilde{x}_0) = Ac(2^{\tilde{Q}}, \Sigma_o, \tilde{\xi}, UR(\{\tilde{q}_0\})).$$

The state estimates of the public under information release described by control policy $\omega = (H, \varphi)$ can be obtained as follows.

Theorem 1: Consider an information release policy $\omega = (H, \varphi)$. After the public observers a string $t \in \theta^{\omega}(\mathcal{L}(G))$, the state estimate of the public is given by

$$SE^{\omega}(t) = \{q \in Q : (\exists y \in Y)(q, y) \in \tilde{\xi}(\tilde{x}_0, t)\}.$$

Example 1:

Let us consider the system modeled by automaton G shown in Figure 2. The initial state is 1. The event set is $\Sigma = \{\alpha, \beta, \gamma\}$ and γ is unobservable (that is, $\Sigma_o = \{\alpha, \beta\}$).

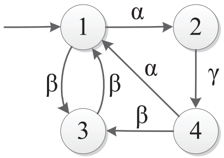

Fig. 2. The system G

We have the following information release policy. Initially, the controller will release the information on the occurrence of β. After the controller observes either α or β, it will release the information as follows. If the last event it observes is α, then it will release the information on the occurrences of both α and β; if the last event observes is β, then it will release the information on the occurrence of only α. This information release policy can be implemented by $\omega = (H, \varphi)$ shown in Figure 3.

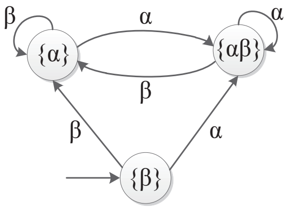

Fig. 3. The information release policy $\omega = (H, \varphi)$

$\tilde{G}$ can be obtained by parallel composition of G and H as shown in Figure 4. The states of $\tilde{G}$ are pairs $\tilde{q} = (q, y)$. The set of events that the public can observe at state $\tilde{q}$ is given by $\varphi(y)$. For instance, in state $(3, \{\alpha\})$, the information on the occurrence of α is released, but not the information on the occurrence of β. Therefore, the transitions $((3, \{\alpha\}), \beta, (1, \{\alpha\}))$ will be replaced by an ε-transition in $\tilde{G}_{\varepsilon}$. Similarly, the following two transitions will also be replaced by ε-transition in $\tilde{G}_{\varepsilon}$: $((1, \{\beta\}), \alpha, (2, \{\alpha\beta\}))$ and $((2, \{\alpha\beta\}), \gamma, (4, \{\alpha\beta\}))$ (the event γ is not observable).

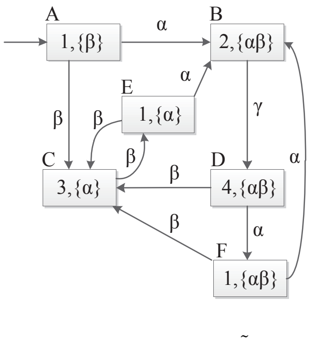

Fig. 4. Parallel composition $\tilde{G} = G||H$

Renaming the states as A, B, C, D, E, F and converting $\tilde{G}_\varepsilon$ into a deterministic automaton, we obtained $\tilde{G}_{obs}$ shown in Figure 5, which gives us the state estimates of the public after observing a string. For instance, after observing $\beta\alpha$, the state estimate is $\{2, 4\}$; and after observing $\beta\alpha\alpha$, the state estimate is $\{1\}$.

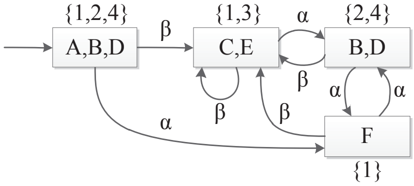

Fig. 5. Observer $\tilde{G}_{obs} = (\tilde{X}, \Sigma_o, \tilde{\xi}, \tilde{x}_0)$

Using observer $\tilde{G}_{obs}$, we can check whether a control policy for information release ensures strong opacity as stated in the following theorem.

Theorem 2: Consider an information release policy $\omega = (H, \varphi)$. The corresponding observation mapping θ^ω ensures strong opacity, that is, $\theta^\omega(L) \subseteq \theta^\omega(K)$, if and only if in the observer $\tilde{G}_{obs} = (\tilde{X}, \Sigma_o, \tilde{\xi}, \tilde{x}_0)$,

$$\begin{aligned}(\forall \tilde{x} \in \tilde{X})\{q \in Q : (\exists y \in Y)(q, y) \in \tilde{x}\} \cap Q_L \neq \emptyset \\ \Rightarrow \{q \in Q : (\exists y \in Y)(q, y) \in \tilde{x}\} \cap Q_K \neq \emptyset.\end{aligned}$$

IV. Off-line Algorithm for Maximum Information Release for Strong Opacity

In the previous section, we investigate how to check whether a given control ω ensures strong opacity or not. In this section, we investigate how to design a control policy ω for maximum information release while ensures strong opacity using an off-line approach.

To do this, we need to make a further assumption on control policy ω. We assume that ω is state-estimate-based, that is, ω depends only on the current state estimate. Formally, state-estimate-based control policy means that $H = G_{obs}$, that is, ω is implemented by (G_{obs}, φ) for some φ.

For two feedback mapping φ_1 and φ_2, define $\varphi_1 \leq \varphi_2$ and $\varphi_1 < \varphi_2$ in the way similar to that for $\omega_1 \leq \omega_2$ and $\omega_1 < \omega_2$. It is not difficult to show that for $\omega_i = (G_{obs}, \varphi_i)$, $i = 1, 2$,

$$\begin{aligned}\omega_1 \leq \omega_2 \Leftrightarrow \varphi_1 \leq \varphi_2 \\ \omega_1 < \omega_2 \Leftrightarrow \varphi_1 < \varphi_2.\end{aligned}$$

For a control policy implemented by (G_{obs}, φ), we can check whether it ensures strong opacity using Theorem 2. However, since the control policy is state-estimate-based, there is a simpler way to do the checking, which is as follows. First, we replace the transitions in G_{obs} that cannot be observed by the public by ε :

$$G_{obs,\varepsilon} = (X, \Sigma_o, \xi_\varepsilon, x_0),$$

where

$$\begin{aligned}\xi_\varepsilon = &\{(x, \sigma, \xi(x, \sigma)) : (x, \sigma, \xi(x, \sigma)) \in \xi \wedge \sigma \in \varphi(x)\} \\ &\cup \{(x, \varepsilon, \xi(x, \sigma)) : (x, \sigma, \xi(x, \sigma)) \in \xi \wedge \sigma \notin \varphi(x)\}.\end{aligned}$$

$G_{obs,\varepsilon}$ is a non-deterministic automaton with ε-transitions. Convert it to a deterministic automaton in the usual way:

$$(G_{obs,\varepsilon})_{obs} = (Z, \Sigma_o, \rho, z_0) = Ac(2^X, \Sigma_o, \rho, UR(x_0)).$$

Then we have the following theorem.

Theorem 3: For a control policy implemented by (G_{obs}, φ), automaton $(G_{obs,\varepsilon})_{obs}$ is isomorphic to automaton $\tilde{G}_{obs}$, denoted by

$$(G_{obs,\varepsilon})_{obs} = \tilde{G}_{obs}.$$

We can now derive the following simpler condition for an state-estimate-based information release policy to ensure strong opacity.

Theorem 4: Consider an state-estimate-based information release policy $\omega = (G_{obs}, \varphi)$. The corresponding observation mapping θ^ω ensures strong opacity, that is, $\theta^\omega(L) \subseteq \theta^\omega(K)$, if and only if in the observer $(G_{obs,\varepsilon})_{obs} = (Z, \Sigma_o, \rho, z_0)$,

$$(\forall z \in Z)(\bigcup_{x \in z} x) \cap Q_L \neq \emptyset \Rightarrow (\bigcup_{x \in z} x) \cap Q_K \neq \emptyset.$$

Based on the above results, the following algorithm can be used to calculate a control that solves the Maximum Information Release Problem for Strong Opacity.

Algorithm 1 (Off-line Algorithm for Strong Opacity)

```
Input: $G$, $\Sigma_o$;
Output: $H$, $\varphi$;
1: $G_{obs} = (X, \Sigma_o, \xi, x_0) = Ac(2^Q, \Sigma_o, \xi, UR(\{q_0\}))$
   with $\xi(x, \sigma) = UR(\{q \in Q : (\exists q' \in x) q \in f(q', \sigma)\})$;
2: $H = G_{obs}$;
3: for all $x \in X$, $\varphi(x) = \emptyset$;
4: for all $x \in X$, for all $\sigma \in \Sigma_o$, do begin
   4.1: $\varphi(x) = \varphi(x) \cup \{\sigma\}$;
   4.2: $G_{obs,\varepsilon} = (X, \Sigma_o, \xi_\varepsilon, x_0)$ with
        $\xi_\varepsilon = \{(x, \sigma, \xi(x, \sigma)) : (x, \sigma, \xi(x, \sigma)) \in \xi \wedge \sigma \in \varphi(x)\}$
        $\cup \{(x, \varepsilon, \xi(x, \sigma)) : (x, \sigma, \xi(x, \sigma)) \in \xi \wedge \sigma \notin \varphi(x)\}$
   4.3: $(G_{obs,\varepsilon})_{obs} = (Z, \Sigma_o, \rho, z_0) =$
        $Ac(2^X, \Sigma_o, \rho, UR(x_0))$ with
        $\rho(z, \sigma) = UR(\{x \in X : (\exists x' \in z) q \in \xi_\varepsilon(x', \sigma)\})$
   4.4: if $(\forall z \in Z)(\bigcup_{x \in z} x) \cap Q_L \neq \emptyset \Rightarrow (\bigcup_{x \in z} x) \cap Q_K \neq \emptyset$
        is not true, then $\varphi(x) = \varphi(x) - \{\sigma\}$
   4.5: End
5: End.
```

Since the computation of observer is of exponential complexity and it is done twice, the computational complexity of Algorithm 1 is double exponential.

V. Maximum Information Release for Weak Opacity

Starting from this section, we consider weak opacity. Therefore, the objective of the controller is to release as much information as possible to the public under the constraint that L is weakly opaque with respect to K. Formally, we want to solve the following problem.

Maximum Information Release Problem for Weak Opacity (MIRPWO)

Find a control ω of information release that satisfies the following two conditions.

1) The observation mapping θ^ω corresponding to ω ensures weak opacity, that is,
$$\theta^\omega(L) \cap \theta^\omega(K) \neq \emptyset.$$
2) For any other control ω' such that $\omega < \omega'$, the corresponding observation mapping $\theta^{\omega'}$ does not ensure weak opacity, that is,
$$\theta^{\omega'}(L) \cap \theta^{\omega'}(K) = \emptyset.$$

To make the problem nontrivial, let us assume that if all information is released, then weak opacity is not satisfied, that is,
$$P(L) \cap P(K) = \emptyset.$$

As before, we assume that the control policy ω has a finite implementation (H, φ), that is, $\omega = (H, \varphi)$. By Proposition 2 and Theorem 1, we have the following necessary and sufficient condition for a control policy ω to ensure weak opacity.

Theorem 5: Consider an information release policy $\omega = (H, \varphi)$. The corresponding observation mapping θ^ω ensures weak opacity, that is, $\theta^\omega(L) \cap \theta^\omega(K) \neq \emptyset$, if and only if in the observer $\tilde{G}_{obs} = (\tilde{X}, \Sigma_o, \tilde{\xi}, \tilde{x}_0)$,
$$\begin{aligned}(\exists \tilde{x} \in \tilde{X})\{q \in Q : (\exists y \in Y)(q, y) \in \tilde{x}\} \cap Q_L \neq \emptyset \\ \wedge \{q \in Q : (\exists y \in Y)(q, y) \in \tilde{x}\} \cap Q_K \neq \emptyset.\end{aligned}$$

As in the case of strong opacity, if a control policy is implemented by (G_{obs}, φ), then automaton $(G_{obs,\varepsilon})_{obs}$ and $\tilde{G}_{obs}$ are isomorphic: $(G_{obs,\varepsilon})_{obs} = \tilde{G}_{obs}$. Therefore, we have the following theorem.

Theorem 6: Consider an state-estimate-based information release policy $\omega = (G_{obs}, \varphi)$. The corresponding observation mapping θ^ω ensures weak opacity, that is, $\theta^\omega(L) \cap \theta^\omega(K) \neq \emptyset$, if and only if in the observer $(G_{obs,\varepsilon})_{obs} = (Z, \Sigma_o, \rho, z_0)$,
$$(\exists z \in Z)(\bigcup_{x \in z} x) \cap Q_L \neq \emptyset \wedge (\bigcup_{x \in z} x) \cap Q_K \neq \emptyset.$$

Using Theorem 6, the following algorithm can be derived that calculates a control ω that solves the Maximum Information Release Problem for Weak Opacity.

Algorithm 2 (Off-line Algorithm for Weak Opacity)

```
Input: $G$, $\Sigma_o$;
Output: $H$, $\varphi$;
1: $G_{obs} = (X, \Sigma_o, \xi, x_0) = Ac(2^Q, \Sigma_o, \xi, UR(\{q_0\}))$
   with $\xi(x, \sigma) = UR(\{q \in Q : (\exists q' \in x) q \in f(q', \sigma)\})$;
2: $H = G_{obs}$;
3: for all $x \in X$, $\varphi(x) = \emptyset$;
4: for all $x \in X$, for all $\sigma \in \Sigma_o$, do begin
   4.1: $\varphi(x) = \varphi(x) \cup \{\sigma\}$;
   4.2: $G_{obs,\varepsilon} = (X, \Sigma_o, \xi_\varepsilon, x_0)$with
        $\xi_\varepsilon = \{(x, \sigma, \xi(x, \sigma)) : (x, \sigma, \xi(x, \sigma)) \in \xi \wedge \sigma \in \varphi(x)\}$
        $\cup \{(x, \varepsilon, \xi(x, \sigma)) : (x, \sigma, \xi(x, \sigma)) \in \xi \wedge \sigma \notin \varphi(x)\}$
   4.3: $(G_{obs,\varepsilon})_{obs} = (Z, \Sigma_o, \rho, z_0) =$
        $Ac(2^X, \Sigma_o, \rho, UR(x_0))$with
        $\rho(z, \sigma) = UR(\{x \in X : (\exists x' \in z) q \in \xi_\varepsilon(x', \sigma)\})$
   4.4: if $(\exists z \in Z)(\bigcup_{x \in z} x) \cap Q_L \neq \emptyset \wedge (\bigcup_{x \in z} x) \cap Q_K \neq \emptyset$
        is not true, then $\varphi(x) = \varphi(x) - \{\sigma\}$
   4.5: End
5: End.
```

Same as Algorithm 1, the computational complexity of Algorithm 2 is also double exponential.

References

[1] M. Ben-Kalefa and F. Lin, "Opaque superlanguages and sublanguages in discrete event systems," *Proceedings of the 48th IEEE Conference on Decision and Control*, pp. 199-204, 2009.

[2] M. Ben-Kalefa and F. Lin, "Supervisory control for opacity of discrete event systems," *Proceedings of 49th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1113-1119, 2011.

[3] J. W. Bryans, M. Koutny, L. Mazare and P. Y. A. Ryan, "Opacity generalised to transition systems," *FAST 2005, LNCS 2860*, pp. 81-95, 2006.

[4] F. Cassez, J. Dubreil and H. Marchand, "Dynamic Observers for the Synthesis of Opaque Systems," *ATVA 2009, LNCS 5799*, pp. 352-367, 2009.

[5] F. Cassez, J. Dubreil and H. Marchand, "Synthesis of opaque system with static and dynamic masks," *Formal Methods in System Design*, 40, pp. 88-115, 2012.

[6] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. Journal of Cryptology, 1:6575, 1988.

[7] J. Dubreil, Ph. Darondeau and H. Marchand, "Opacity enforcing control synthesis," *Proceedings of the 9th International Workshop on Discrete Event Systems*, pp. 28-35, 2008.

[8] J. Dubreil, Ph. Darondeau and H. Marchand, "Supervisory control for opacity," *IEEE Transactions on Automatic Control*, 55(5), pp. 1089-1100, 2010.

[9] F. Lin, "Opacity of discrete event systems and its applications," *Automatica, 47,* pp. 496-503, 2011.

[10] F. Lin and W. M. Wonham, "On observability of discrete-event systems," *Information Sciences*, 44, pp. 173-198, 1988.

[11] A. Saboori and C. Hadjicostis, "Notions of security and opacity in discrete event systems," *Proceedings of the 46th IEEE Conference on Decision and Control*, pp. 5056-5061, 2007.

[12] A. Saboori and C. Hadjicostis, "Verification of initial-state opacity in security applications of DES," *Proceedings of the 9th International Workshop on Discrete Event Systems*, pp. 328-333, 2008.

[13] A. Saboori and C. Hadjicostis, "Verification of infinite-step opacity and complexity considerations," *IEEE Transactions on Automatic Control*, 57(5) pp. 1265-1269, 2012.

[14] A. Saboori and C. Hadjicostis, "Opacity-enforcing supervisory strategies via state estimator constructions," *IEEE Transactions on Automatic Control*, 57(5) pp. 1155-1165, 2012.

[15] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Diagnosability of discrete event systems. *IEEE Transactions on Automatic Control, 40(9)*, pp. 1555-1575, 1995.

[16] S. Shu and F. Lin, "Detectability of discrete event systems with dynamic event observation," *Systems & Control Letters, 59(1),* pp. 9-17, 2010.

[17] Y. Wu and S. Lafortune, "Enforcement of opacity properties using insertion functions," *51st IEEE Conference on Decision and Control,* pp. 6722-6728, 2012.

[18] B. Zhang, S. Shu and F. Lin, "Polynomial algorithm to check opacity in discrete event system," *Proceedings of 24th Chinese Control and Decision Conference*, pp. 763-769, 2012.