Supervisory Control for Delayed Detectability of Discrete Event Systems

Xiang Yin and Shaoyuan Li

Abstract— **Detectability is an important property that characterizes whether or not the state of a system can be precisely determined under imperfect observation. In this paper, we investigate the supervisor synthesis problem for a specific class of detectability, called delayed detectability, in the context of Discrete Event Systems. Specifically, delayed detectability requires that the state of the system can always be determined after some bounded information delays. That is, one is allowed to use future information to "smooth" the state estimate of previous instant. Our goal is to design a maximally-permissive supervisor that restricts the behavior of the original system such that the closed-loop system under control is delayed detectable. To this end, we propose a new supervisor synthesis approach that appropriately handles the delayed information in the control problem . Our work generalizes previous results on supervisory control of detectability, where only non-delayed information is considered.**

I. Introduction

State estimation and detection are central problems in the systems theory. In most of the real-world systems, the state of the system cannot be perfectly measured due to measurement noises or measurement uncertainties. Therefore, it is necessary to perform state estimation and detection algorithms to obtain more precise knowledge of the system. State estimation and detection are useful for many practical purposes, e.g., supervisory control, fault diagnosis, fault prognosis and security analysis.

In the DES literature, the study of the state estimation problem dates back to [9], [10], where the notion of observability is investigated. More recently, Shu and Lin investigated this problem more systematically and proposed the concept of *detectability*. In [15], four different notions of detectability were proposed including (periodically) strong detectability and (periodically) weak detectability. Since then, detectability has drawn increasing attention in the past few years; see, e.g., [1], [3], [6], [8], [8], [12]–[14], [17], [19], [21], [22], [24], [26], [26], [27]. For example, in [4], [5], [16], [18], [28], the authors extended detectability to the stochastic setting by explicitly considering the probability of detection. In [8], [17], [27], detectability has been extended to systems modeled as labeled Petri nets. In [6], [7], [26], the complexity of checking different notions of detectability has been investigated.

Most of the state detection problems in the literature focus on the detection of the current state of the system. However, in some situations, we are able to use *future information* to improve our knowledge of the system's state at some previous instant [11], [23], [25]. This is also referred to as the "smoothing" process in the systems theory. Essentially, by using future information, we are able to eliminate the possibility of those states that do not have the observed future behavior from the state estimate in order to enhance precision of the state estimation for a previous instant. In order to utilize the delayed information in the state detection problem, in [13], Shu and Lin proposed a new type of detectability called *delayed detectability*. Specifically, delayed detectability requires that, after k_1 steps, we can always unambiguously determine the precise state of the system with at most k_2 steps of delay. More recently, delayed detectability has been extended to the nondeterministic observation setting by [29].

In many applications, however, the open-loop system may not be detectable directly. Therefore, in order to obtain better information about the system, it is important to *enforce* detectability via some enforcement mechanisms. One widely used property enforcement mechanism in the DES literature is the *supervisory control theory* initiated by Ramadge and Wonham [2]. The supervisory control approach has already been applied in the literature to enforce detectability; e.g., [14], [19]. For example, in [14], the authors show how to design a supervisor that enforces strong detectability; it has been pointed out that maximally-permissive supervisor does not exist in general for this problem. In [19], the authors study the enforcement of strong K-detectability using supervisory control, where maximally-permissive supervisor does exists.

In this paper, we investigate how to synthesize a supervisor that enforces detectability when delayed information is involved. Specifically, we study the enforcement of (k_1, k_2)-detectability proposed by [13]. To this end, we propose a new structure called the (k_1, k_2)-observer that precisely captures the delayed information in the synthesis problem. An effective approach is provided to synthesize an detectability enforcing supervisor for this case. In particular, we show that a maximally-permissive supervisor always exists and it is unique, i.e., supremal, when all controllable events are observable.

II. Preliminaries

A. System Model

Let Σ be a finite set of events. A string is a finite sequence of events and a language is a set of strings. We denote by Σ^* the set of all strings over Σ including the empty string ϵ. For any string $s \in \Sigma^*$, we denote by $|s|$ its length with $|\epsilon| = 0$; we denote by σ_i^s the ith event in s, i.e., $s = \sigma_1^s \sigma_2^s \dots \sigma_{|s|}^s$.

This work was supported by the National Natural Science Foundation of China (61803259, 61833012).

Xiang Yin and Shaoyuan Li are with Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China. {yinxiang,syli}@sjtu.edu.cn.

We consider a DES modeled by a finite-state automaton

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

where X is a finite set of states, Σ is a finite set of events, $\delta : X \times \Sigma \to 2^X$ is a non-deterministic partial transition function and $X_0 \subseteq X$ is the set of initial states. The transition function is also extended to $\delta : X \times \Sigma^* \to 2^X$ recursively by: $\forall s \in \Sigma^*, \sigma \in \Sigma, x \in X : \delta(x, s\sigma) = \cup_{x' \in \delta(x,s)} \delta(x', \sigma)$. The language generated by G from state $x \in X$ is defined by $\mathcal{L}(G, x) := \{s \in \Sigma^* : \delta(x, s)!\}$, where "!" means "is defined". We define $\mathcal{L}(G) := \cup_{x_0 \in X_0} \mathcal{L}(G, x_0)$ as the language generated by the system.

For any finite-state automaton $G = (X, \Sigma, \delta, X_0)$, we denote by $G_R = (X, \Sigma, \delta_R, X)$ the *reversed automaton* of G. Specifically, the transition function $\delta_R : X \times \Sigma \to 2^X$ is defined by: $\forall x, x' \in X, \sigma \in \Sigma : x' \in \delta_R(x, \sigma) \Leftrightarrow x \in \delta(x', \sigma)$. Note that the initial state of G_R is the entire state space. For any string $s \in \Sigma^*$, we denote by s_R its reversed string, i.e., $s_R = \sigma^s_{|s|} \dots \sigma^s_2 \sigma^s_1$.

We assume that the system G is partially-observed. Specifically, we assume that the event set is partitioned as $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, where Σ_o is the set of observable events and Σ_{uo} is the set of unobservable events. The natural projection from $P : \Sigma^* \to \Sigma_o^*$ is defined recursively by: $\forall s \in \Sigma^*, \sigma \in \Sigma$

$$P_(\epsilon) = \epsilon, \quad P(s\sigma) = \begin{cases} P(s)\sigma & \text{if } \sigma \in \Sigma_o \\ P(s) & \text{if } \sigma \in \Sigma_{uo} \end{cases}$$

The natural projection is also extended to $P : 2^{\Sigma^*} \to 2^{\Sigma_o^*}$ by: $\forall L \subseteq \Sigma^* : P(L) = \{P(s) \in \Sigma_o^* : s \in L\}$.

B. Delayed Detectability

Let G be a system and $\alpha\beta \in P(\mathcal{L}(G))$ be an observable string. We denote by $\hat{X}(\alpha \mid \alpha\beta)$ the set of states the system could have been in for the instant when α is observed given the entire observation $\alpha\beta$, i.e.,

$$\hat{X}_G(\alpha \mid \alpha\beta) := \left\{ x \in X : \begin{array}{c} \exists x_0 \in X_0, s_1 s_2 \in \mathcal{L}(G, x_0) \text{ s.t.} \\ P(s_1) = \alpha \wedge P(s_1 s_2) = \alpha\beta \\ \wedge x \in \delta(x_0, s_1) \end{array} \right\}$$

We define $\hat{X}_G(\alpha) := \hat{X}_G(\alpha \mid \alpha)$. Therefore, $\hat{X}(\alpha \mid \alpha\beta)$ and $\hat{X}_G(\alpha)$ are also referred to as the *delayed state estimate* and the *current state estimate*, respectively.

The current-state estimate $\hat{X}_G(\alpha)$ can be computed by the well-known *observer automaton*. Let $q \in 2^X$ be a set of states and $\sigma \in \Sigma_o$ be an observable event. Then we define

$$UR(q) := \{x' \in X : \exists x \in q, w \in \Sigma_{uo}^* \text{ s.t. } x' \in \delta(x, w)\}$$

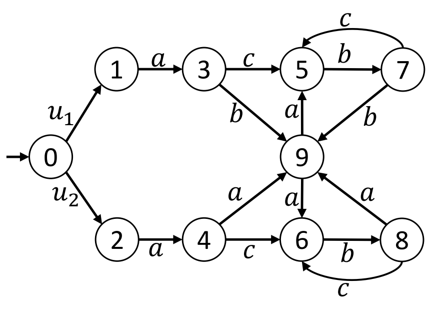
as the unobservable reach of q and define

$$Next_\sigma(q) := \{x' \in X : \exists x \in q \text{ s.t. } x' \in \delta(x, \sigma)\}$$

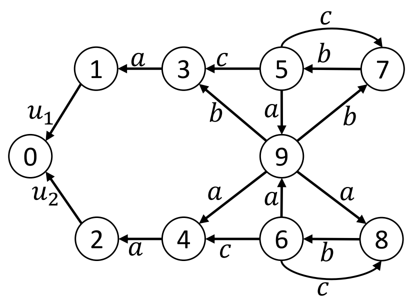
as the observable reach of q upon the occurrence of σ.

To compute the current-state estimate, we can construct the observer. Specifically, the observer of G is a finite-state automaton

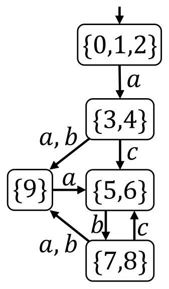
$$Obs(G) = (X_{obs}, \Sigma_o, \delta_{obs}, x_{obs,0}),$$



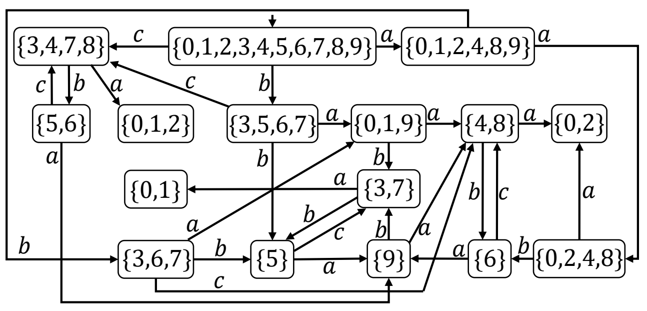
(a) G



(b) G_R



(c) $Obs(G)$



(d) $Obs(G_R)$

Fig. 1. System G with $\Sigma_o = \{a, b, c\}$ and $\Sigma_c = \{a, c\}$. Note that all states are initial in G_R.

where $X_{obs} \subseteq 2^X \setminus \emptyset$, $x_{obs,0} = UR(X_0)$ and $\delta_{obs} : X_{obs} \times \Sigma_o \to X_{obs}$ is a deterministic transition function defined by: $\forall q \in X_{obs}, \sigma \in \Sigma_o : \delta(q, \sigma) = UR(Next_\sigma(q))$. Then we have $\mathcal{L}(Obs(G)) = P(\mathcal{L}(G))$ and for any $\alpha \in P(\mathcal{L}(G)) : \hat{X}_G(\alpha) = \delta_{obs}(q_0, \alpha)$, which computes the current-state estimate upon the occurrence of α.

To compute the delayed-state estimate, one can construct both the observer of the original system and the observer of the reversed system, i.e., $Obs(G)$ and $Obs(G_R)$. It has been shown in [20] that, for any string $\alpha\beta \in P(\mathcal{L}(G))$, we can compute the delayed-state estimate by

$$\hat{X}_G(\alpha \mid \alpha\beta) = \hat{X}_G(\alpha) \cap \hat{X}_{G_R}(\beta_R). \tag{1}$$

Note that $\hat{X}_{G_R}(\beta_R)$ is the state reached via β_R in $Obs(G_R)$.

Example 1: Let us consider system G shown in Figure 1(a) with $\Sigma_o = \{a, b, c\}$. Its reversed automaton G_R is shown in Figure 1(b), where all states are initial. Then $Obs(G)$ and $Obs(G_R)$ are shown in Figures 1(c) and 1(d), respectively. For example, when we observe string $ac \in$

$P(\mathcal{L}(G))$, we have $\hat{X}_G(ac) = \delta_{obs}(q_0, ac) = \{5,6\}$. If we observe string ba after observing ac, then we have

$$\hat{X}_G(ac \mid acba) = \hat{X}_G(ac) \cap \hat{X}_{G_R}(ab) = \{5,6\} \cap \{3,6,7\} = \{6\}$$

i.e., the system must be at state 6 two steps ago when $acba$ is observed. ■

C. Delayed Detectability

In the state detection problem, the goal is to precisely determine the state of the system, possibly with some information delay. To this end, in [13], Shu and Lin propose the notion of (k_1, k_2)-detectability, which is reviewed as follows.

Definition 1: ((k_1,k_2)-Detectability) Let $k_1, k_2 \in \mathbb{N}$ be two non-negative integers. System G is said to be (k_1,k_2)-detectable w.r.t. Σ_o if, for any $\alpha\beta \in P(\mathcal{L}(G))$ such that $|\alpha| \geq k_1$ and $|\beta| \geq k_2$, we have $|\hat{X}_G(\alpha \mid \alpha\beta)| = 1$.

Intuitively, (k_1,k_2)-detectability says that, after k_1 (observational) steps from the initial state, any state of the system can be precisely determined within at most k_2 steps. This definition is weaker than strong (current-state) detectability [15] since we can use future information to improve our knowledge of the system for some previous instant.

Example 2: We still consider system G shown in Figure 1(a) with $\Sigma_o = \{a,b,c\}$. This system is not $(1,n)$-detectable for any n, since $ac(bc)^n \in P(\mathcal{L}(G))$ and $\hat{X}_G(a \mid ac(bc)^n) = \{3,4\}$. That is, we may never be able to determine the precise state of the system for the instant of the occurrence of a within any finite delay. ■

III. Supervisory Control for Delayed Detectability

A. Problem Formulation

In many situations, the open-loop system G may not be detectable. Therefore, we need to enforce detectability via some mechanism in order to maintain precise knowledge of the system. One of the most widely used enforcement mechanisms is the supervisory control theory initiated by Ramadge and Wonham. In this framework, we assume that the event set is further partitioned as $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc}$, where Σ_c and Σ_{uc} are the set of controllable events and the set of uncontrollable events, respectively. We assume that $\Sigma_c \subseteq \Sigma_o$, i.e., we can only disable observable events which is the case in many applications.

A control decision is a set of events that includes Σ_{uc}; we denote by $\Gamma = \{\gamma \in 2^\Sigma : \Sigma_{uc} \subseteq \gamma\}$ the set of control decisions. Then a *partial-observation supervisor* is a function

$$S : \Sigma_o^* \to \Gamma,$$

i.e., for any observable string $\alpha \in P(\mathcal{L}(G))$, $S(\alpha)$ is the set of events enabled upon the observation of α. We denote by S/G the closed-loop system under control. The language generated by the closed-loop system $\mathcal{L}(S/G)$ is defined recursively by:

- $\epsilon \in \mathcal{L}(S/G)$; and
- $\forall s \in \Sigma^*, \sigma \in \Sigma : s\sigma \in \mathcal{L}(S/G) \Leftrightarrow s \in \mathcal{L}(S/G) \wedge \sigma \in S(P(s)) \wedge s\sigma \in \mathcal{L}(G)$;

In the closed-loop system under control, since some possible string can be disabled by the supervisor, the information of the system may be changed under supervision. Therefore, we define

$$\hat{X}_{S/G}(\alpha \mid \alpha\beta) := \left\{ x \in X : \begin{array}{r} \exists x_0 \in X_0, \exists s_1 s_2 \in \mathcal{L}(S/G) \text{ s.t.} \\ P(s_1) = \alpha \wedge P(s_1 s_2) = \alpha\beta \\ \wedge x \in \delta(x_0, s_1) \end{array} \right\}$$

as the delayed state estimate under control. Compared with $\hat{X}_G(\alpha \mid \alpha\beta)$, $\hat{X}_{S/G}(\alpha \mid \alpha\beta)$ only consider observation consistent strings in $\mathcal{L}(S/G)$.

The detectability enforcement problem is to synthesize a supervisor such that the closed-loop system is detectable. This problem is formulated as follows.

Problem 1: Given system G with $\Sigma_c \subseteq \Sigma_o \subseteq \Sigma$, synthesize a supervisor $S : P(\mathcal{L}(G)) \to \Gamma$ such that, for any $\alpha\beta \in P(\mathcal{L}(S/G)) : |\alpha| \geq k_1, |\beta| \geq k_2$, we have $|\hat{X}_{S/G}(\alpha \mid \alpha\beta)| = 1$.

We say that a solution S to Problem 1 is *optimal* if for any other solution S', we have $\mathcal{L}(S'/G) \subseteq \mathcal{L}(S/G)$.

B. Separability in Delayed State Estimation

In order to solve Problem 1, we need to provide an effective approach to compute the closed-loop delayed state estimate $\hat{X}_{S/G}(\alpha \mid \alpha\beta)$. Note that, in general, we always have $\hat{X}_{S/G}(\alpha \mid \alpha\beta) \subseteq \hat{X}_G(\alpha \mid \alpha\beta)$.

In [19], the authors have shown that, under the assumption of $\Sigma_c \subseteq \Sigma_o$, the current state estimates of the open-loop system and the closed-loop system are the same, i.e., $\hat{X}_{S/G}(\alpha) = \hat{X}_G(\alpha)$ for any supervisor S. Therefore, the state estimation and the control synthesis can be separated. Here, we further generalize this separability result to the case of delayed state estimate.

Proposition 1: Assume that $\Sigma_c \subseteq \Sigma_o$. For any supervisor $S : P(\mathcal{L}(G)) \to \Gamma$ and any string $\alpha\beta \in P(\mathcal{L}(S/G))$, we have $\hat{X}_{S/G}(\alpha \mid \alpha\beta) = \hat{X}_G(\alpha \mid \alpha\beta)$.

IV. The (k_1, k_2)-Observer

In this section, we present the structure of the (k_1,k_2)-observer, which generalizes the standard observer structure to handle delayed information. This structure will be used as the basis for solving the control synthesis problem.

Before, we formally present the definition of the (k_1,k_2)-observer, we first introduce some necessary notations.

For any non-negative integer $n \in \mathbb{N}$, we define

$$\Sigma^{\leq n} := \{s \in \Sigma^* : |s| \leq n\}$$

as the set of strings whose lengths are smaller than or equal to n. Also, for any string $s = \sigma_1^s \sigma_2^s \ldots \sigma_{|s|}^s \in \Sigma^*$ and non-negative integer $n \in \mathbb{N}$, we denote by $Last_n(s)$ the string consisting of the last n events of s, i.e.,

$$Last_n(s) = \begin{cases} s & \text{if } |s| \leq n \\ \sigma_{|s|-n+1}^s \cdots \sigma_{|s|}^s & \text{if } |s| > n \end{cases}$$

We are now ready to define the (k_1,k_2)-observer.

Given system G with $\Sigma_o \subseteq \Sigma$ and two non-negative integers $k_1, k_2 \in \mathbb{N}$, the (k_1, k_2)-observer for G is defined as a new finite-state automaton

$$Obs_{k_1,k_2}(G) = (Q, \Sigma_o, f, q_0)$$

where

- $Q \subseteq (2^X \setminus \emptyset) \times \Sigma^{\leq k_2} \times \{0, 1, \ldots, k_1\}$ is the set of states;
- Σ_o is the set of observable events;
- $f : Q \times \Sigma_o \to Q$ is the deterministic partial transition function defined as follows:
  - For any state $\imath = (q, s, n)$ and any observable event $\sigma \in \Sigma_o$, $f(\imath, \sigma)$ is defined if and only if $\delta_{obs}(q, s\sigma)$ is defined.
  - Moreover, if $f((q, s, n), \sigma)$ is defined, then it leads to state $\imath' = (q', s', n')$ defined by:
    - If $n < k_1$, then
    $$(q', s', n') = (\delta_{obs}(q, \sigma), \epsilon, n + 1) \quad (2)$$
    - If $n = k_1$ and $|s| < k_2$, then
    $$(q', s', n') = (q, s\sigma, k_1) \quad (3)$$
    - If $n = k_1$ and $|s| = k_2$, then
    $$(q', s', n') = (\delta_{obs}(q, \sigma_1^s), Last_{k_2}(s\sigma), k_1) \quad (4)$$
- $q_0 := (UR(X_0), \epsilon, 0) \in Q$ is the initial state.

For the sake of simplicity, we only consider the reachable part of the (k_1, k_2)-observer.

The (k_1, k_2)-observer is the key information structure in this paper. Essentially, it divides the evolution of the system into three phases:

- the *transient phase*: less than k_1 observable events are generated; and
- the *event-delay phase*: more than k_1 but less than k_2 observable events are generated;
- the *evaluation phase*: more than $k_1 + k_2$ observable events are generated.

More intuitively, each phase works as follows.

- We say that system is at the transient phase when less than k_1 observable events are executed, since we do not need to determine the precise state of the system for these instants. Therefore, the first component simply tracks the current state estimate following the dynamic of the system, and the second component will not store any string information. The last integer component is simply a counter that determines whether or not the system is at the transient phase, i.e., how many observable events have occurred from the initial state. Therefore, upon the occurrence of each new observable event, its value is added by one and it will only record at most k_1 steps.
- We say that system is at the event-delay phase when more than k_1 but less than $k_1 + k_2$ observable events are executed, since we need to determine the precise state of the system for these instants, but not for now, i.e., the detection bound k_2 has not expired. Therefore, we will not update the state estimate immediately. Instead, we will keep the first component unchanged and store the observable string following the receiving order until there are k_2 events in the string. Note that the last component will not change anymore since the transient phase has expired.
- Finally, the system goes to the evaluation phase after $k_1 + k_2$ observable events are generated. This is because we need to evaluate whether or not the state of the instant k_2 steps ago can be determined. In this phase, the string in the second component is used to record the last k_2 observable events after k_1 observable events have occurred. Therefore, whenever a new event is observed, the last event will be erased by the $Last_{k_2}$ operator and the new event will be added to the rear of the string; the string in this component is essentially a *queue*. Hence, for any $\alpha = \sigma_1^\alpha \ldots \sigma_{|\alpha|}^\alpha$ such that$(q, s, k) = f(q_0, \alpha)$, we have
$$s = \begin{cases} \epsilon & \text{if } |\alpha| \leq k_1 \\ Last_{k_2}(\sigma_{k_1+1}^\alpha \ldots \sigma_{|\alpha|}^\alpha) & \text{if } |\alpha| > k_1 \end{cases} \quad (5)$$

Then the state estimate component is updated based on the event erased at each instant, i.e., σ_1^s. Therefore, each state at the phase contains the current state estimate for the instant k_2 steps ago and the string with length k_2 that is executed from that instant. Formally, for any $\alpha = \sigma_1^\alpha \ldots \sigma_{|\alpha|}^\alpha$ such that $f(q_0, \alpha) = (q, s, n)$, we have

$$q = \begin{cases} \hat{X}_G(\alpha) & \text{if } |\alpha| < k_1 \\ \hat{X}_G(\sigma_1^\alpha \ldots \sigma_{k_1}^\alpha) & \text{if } k_1 \leq |\alpha| < k_1 + k_2 \\ \hat{X}_G(\sigma_1^\alpha \ldots \sigma_{|\alpha|-k_2}^\alpha) & \text{if } k_1 + k_2 \leq |\alpha| \end{cases} \quad (6)$$

Finally, we note that the transition function of the (k_1, k_2)-observer is defined if and only if the transition function of the standard observer is defined. Therefore, we also have $\mathcal{L}(Obs_{k_1,k_2}(G)) = \mathcal{L}(Obs(G)) = P(\mathcal{L}(G))$.

Let us illustrate this structure by the following example.

Example 3: Still, let us consider system G shown in Figure 1(a). We consider $k_1 = 1$ and $k_2 = 2$. Then the (k_1, k_2)-observer $Obs_{1,2}(G)$ is shown in Figure 2.

Initially, $Obs_{1,2}(G)$ starts from state $(\{0, 1, 2\}, \epsilon, 0)$, where $\{0, 1, 2\}$ is the initial state of $Obs(G)$ and the string queue is empty. Upon the occurrence of the first observable event a, the system moves to state $(\{3, 4\}, \epsilon, 1)$. Specifically, since $0 < k_1 = 1$, the observer is still at the *transient phase*, i.e., (i) $\{3, 4\}$ is computed by updating the current-state estimate $\{0, 1, 2\}$ using event a; and (ii) the string queue remains as the empty string; and (iii) the integer component is added to 1.

When state $(\{3, 4\}, \epsilon, 1)$ is reached, the observer becomes to the *event-delay phase*, since $1 = k_1$ but $|\epsilon| < k_2$. Therefore, if event c is observed, then we move to state $(\{3, 4\}, c, 1)$, where (i) state estimate $\{3, 4\}$ is not changed; and (ii) the string queue becomes to c by adding event c to the queue; and (iii) the integer component is not changed anymore by remaining as $1 = k_1$.

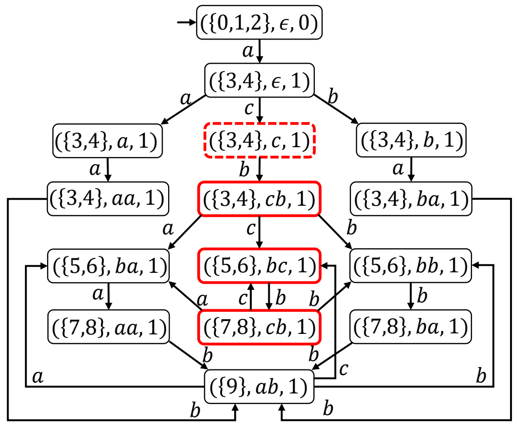


Fig. 2. The (k_1, k_2)-observer $Obs_{k_1,k_2}(G)$ for system G shown in Figure 1(a), where $k_1 = 1$ and $k_2 = 2$.

When state $(\{3,4\}, cb, 1)$ is reached, the observer becomes to the *evaluation phase*, since $|cb| = 2 = k_2$. If event a is observed from state $(\{3,4\}, cb, 1)$, then state $(\{5,6\}, ba, 1)$ is reached. Specifically, the state estimate $\{3,4\}$ is updated to $\{5,6\}$ by using the first event c in the queue. Therefore, event c is consumed and we add new event a to the end of the queue, which gives ba. Still, the integer component is not changed anymore by remaining as $1 = k_1$.

V. Synthesis Procedure

In order to enforce delayed detectability, we need to avoid the occurrence of observable string $\alpha\beta \in P(\mathcal{L}(G))$ such that $|\alpha| \geq k_1, |\beta| \geq k_2$ and $|\hat{X}_G(\alpha \mid \alpha\beta)| > 1$. Since the delayed state estimate for the instant of α can only be improved when β goes longer, it suffices to consider the case of $|\beta| = k_2$. As we discussed in Equations (5) and (6), for any $\alpha = \sigma_1^\alpha \ldots \sigma_{|\alpha|}^\alpha$ such that $|\alpha| \geq k_1 + k_2$ and $f(q_0, \alpha) = (q, s, n)$, we have

- $q = \hat{X}_G(\sigma_1^\alpha \ldots \sigma_{|\alpha|-k_2}^\alpha)$; and
- $s = Last_{k_2}(\alpha) = \sigma_{|\alpha|-k_2+1}^\alpha \ldots \sigma_{|\alpha|}^\alpha$.

According to Equation (1), we can actually use the information of q and s to reconstruct $\hat{X}_G(\sigma_1^\alpha \ldots \sigma_{|\alpha|-k_2}^\alpha \mid \alpha)$ by

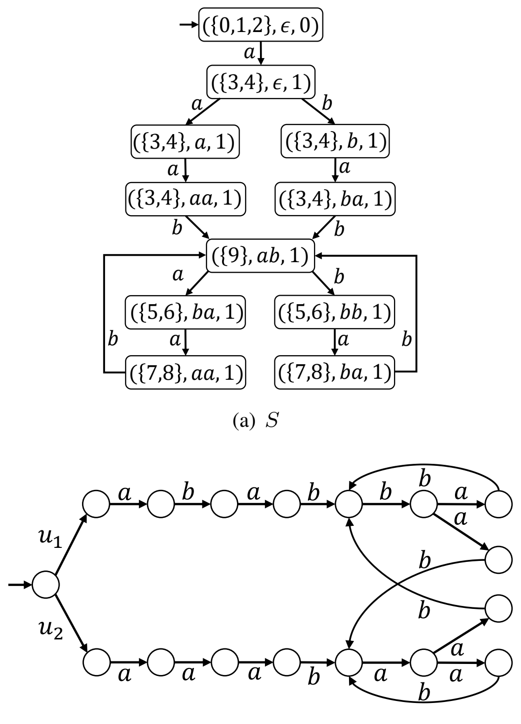
$$\begin{aligned}
&\hat{X}_G(\sigma_1^\alpha \ldots \sigma_{|\alpha|-k_2}^\alpha \mid \alpha)\\
=&\hat{X}_G(\sigma_1^\alpha \ldots \sigma_{|\alpha|-k_2}^\alpha) \cap \hat{X}_{G_R}((Last_{k_2}(\alpha))_R) \qquad (7)\\
=&q \cap \hat{X}_{G_R}(s_R)
\end{aligned}$$

Based on this intuition, we define the set of illegal states in $Obs_{k_1,k_2}(G)$ as follows.

Definition 2: Let $Obs_{k_1,k_2}(G) = (Q, \Sigma_o, f, q_0)$ be the (k_1, k_2)-observer of G. For any state $\imath = (q, s, n) \in Q$, we say that $\imath$ is an *illegal state* if (i) $n = k_1$; and (ii) $|s| = k_2$; and (iii) $|q \cap \hat{X}_{G_R}(s_R)| > 1$. We denote by $Q_{illegal} \subseteq Q$ the set of illegal states in $Obs_{k_1,k_2}(G)$.

The following result reveals that a supervisor S enforces (k_1, k_2)-detectability if and only if it avoids the occurrences of strings leading to illegal states in $Obs_{k_1,k_2}(G)$.

Theorem 1: Supervisor S solves Problem 1 if and only if $\forall \alpha \in P(\mathcal{L}(S/G)), f(q_0, \alpha) \notin X_{illegal}$.



(b) The closed-loop language under control $\mathcal{L}(S\|G)$.

Fig. 3. System G with $\Sigma_o = \{a, b, c\}$ and $\Sigma_c = \{a, c\}$. Note that all states are initial in G_R.

Based on the Theorem 1, we are able to find an optimal solution to Problem 1. Specifically, we need to first build the (k_1, k_2)-observer $Obs_{k_1,k_2}(G)$. Then we just need to solve a standard fully observed supervisory control problem by considering $Obs_{k_1,k_2}(G)$ as the system model, Σ_c as the set of controllable events and $Q_{illegal}$ as the set of illegal states in $Obs_{k_1,k_2}(G)$. The reader is referred to [2] for more details on the standard supervisory control algorithm under the full observation setting. Essentially, it performs an fixed-point computation by iteratively removing illegal states and states with uncontrollable out-going transitions from $Obs_{k_1,k_2}(G)$. The synthesis procedure resulting a sub-automaton of $Obs_{k_1,k_2}(G)$, say S, which can be used as the optimal (maximally-permissive) supervisor. The closed-loop system can be represented as $S\|G$, where "$\|$" denotes the standard parallel composition operator (see, e.g., p. 80 of [2])[1].

We illustrate the synthesis procedure by the following example.

Example 4: We still consider our running example system G shown in Figure 1(a). Its (k_1, k_2)-observer $Obs_{k_1,k_2}(G)$ has been shown in Figure 2 when $k_1 = 1$ and $k_2 = 2$. We can check that states $(\{3,4\}, cb, 1)$, $(\{5,6\}, bc, 1)$ and $(\{7,8\}, cb, 1)$ highlighted by solid red lines are illegal states. For example, for $(\{3,4\}, cb, 1)$, we have $1 = k_1, |cb| = 2 =$

[1]In general, for partially-observation supervisor, we need to first add controllable and unobservable self-loops at each supervisor state, and take the product composition with the original system to get the closed-loop system. However, since we assume here that all controllable events are observable, it suffices to consider the parallel composition directly.

k_2 and

$$\begin{aligned}\{3,4\}\cap \hat{X}_{G_R}((cb)_R) =&\{3,4\}\cap \delta^R_{obs}(X,bc)\\ =&\{3,4\}\cap\{3,4,7,8\}=\{3,4\}\end{aligned}$$

where δ^R_{obs} denotes the transition function of $Obs(G_R)$; recall that the initial state of $Obs(G_R)$ is X. Clearly, $(\{3,4\},cb,1)\in Q_{illegal}$ since $|\{3,4\}\cap \hat{X}_{G_R}((cb)_R)|=2>1$.

In order to solve the standard supervisory control problem based on $Obs_{k_1,k_2}(G),\Sigma_c$ and $Q_{illegal}$, first, we need to remove all illegal states from $Obs_{k_1,k_2}(G)$. However, since state $(\{3,4\},cb,1)$ is removed and event b is an uncontrollable event, state $(\{3,4\},c,1)$, which is highlighted by red dashed line, also needs to be removed in the second iteration. This gives automaton S shown in Figure 3(a), which is the optimal (maximally permissive) supervisor that solves Problem 1. By taking the parallel composition between S and G, we obtain the closed-loop behavior under control, which is shown in Figure 3(b).

Remark 1: Finally, we discuss the computational complexity for solving Problem 1 using the proposed approach. First, we need to construct $Obs_{k_1,k_2}(G)$, which contains at most $k_1\times|\Sigma_o|^{k_2}\times 2^{|X|}$ states and $k_1\times|\Sigma_o|^{k_2+1}\times 2^{|X|}$ transitions. In order to determine $Q_{illegal}$, we need to first construct $Obs(G_R)$, which contains at most $2^{|X|}$ states and $|\Sigma_o|\times 2^{|X|}$ transitions, and then test whether or not $|q\cap \hat{X}_{G_R}(s_R)|>1$ for each (q,s,n) in $Obs_{k_1,k_2}(G)$ such that $|s|=k_2$ and $n=k_1$. Therefore, the entire complexity for determining $Q_{illegal}$ is $O(k_1\times|\Sigma_o|^{k_2}\times 4^{|X|})$. Finally, solving a standard supervisory control problem for state avoidance can be done in linear time in the size of the system. Therefore, the overall complexity is $O(k_1\times|\Sigma_o|^{k_2+1}\times 4^{|X|})$.

VI. Conclusion

In this paper, we solved the supervisory control problem for delayed detectability. The synthesized supervisor guarantees that the state of any instant of the system after k_1 steps can be determined unambiguously with at most k_2 steps of delay. To this end, a new information structure called the (k_1,k_2)-observer is proposed to capture all information needed in the synthesis problem in a finite domain. Based on this structure, we further show that, under the mild assumption of all controllable events are observable, the synthesis problem can be reduced to a standard supervisory control problem under full observation for state avoidance specification.

References

[1] M.V.S. Alves and J.C. Basilio. State estimation and detectability of networked discrete event systems with multi-channel communication networks. In *American Control Conference*, 2019.

[2] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Springer, 2nd edition, 2008.

[3] X. Han, Z. Chen, and J. Zhao. Matrix approach to detectability of discrete event systems under partial observation. In *13th IEEE Conf. Automation Science and Engineering*, pages 187–192, 2017.

[4] C. Keroglou and C.N. Hadjicostis. Detectability in stochastic discrete event systems. *Syst. Control Letters*, 84:21–26, 2015.

[5] C. Keroglou and C.N. Hadjicostis. Verification of detectability in probabilistic finite automata. *Automatica*, 86:192–198, 2017.

[6] T. Masopust. Complexity of deciding detectability in discrete event systems. *Automatica*, 93:257–261, 2018.

[7] T. Masopust and X. Yin. Complexity of detectability, opacity and A-diagnosability for modular discrete event systems. *Automatica*, 101:290–295, 2019.

[8] T. Masopust and X. Yin. Deciding detectability for labeled Petri nets. *Automatica*, 104:238–241, 2019.

[9] C.M. Özveren and A.S. Willsky. Observability of discrete event dynamic systems. *IEEE Trans. Automatic Control*, 35(7):797–806, 1990.

[10] P. Ramadge. Observability of discrete event systems. In *25th IEEE Conf. Decision and Control*, number 25, pages 1108–1112, 1986.

[11] A. Saboori and C.N. Hadjicostis. Verification of k-step opacity and analysis of its complexity. *IEEE Trans. Automation Science and Engineering*, 8(3):549–559, 2011.

[12] S. Shu and F. Lin. Detectability of discrete event systems with dynamic event observation. *Syst. Control Let.*, 59(1):9–17, 2010.

[13] S. Shu and F. Lin. Delayed detectability of discrete event systems. *IEEE Trans. Automatic Control*, 58(4):862–875, 2013.

[14] S. Shu and F. Lin. Enforcing detectability in controlled discrete event systems. *IEEE Trans. Automatic Control*, 58(8):2125–2130, 2013.

[15] S. Shu, F. Lin, and H. Ying. Detectability of discrete event systems. *IEEE Trans. Automatic Control*, 52(12):2356–2359, 2007.

[16] S. Shu, F. Lin, H. Ying, and X. Chen. State estimation and detectability of probabilistic discrete event systems. *Automatica*, 44(12):3054–3060, 2008.

[17] Y. Tong and H. Lan. Verification of detectability in labeled Petri nets. In *American Control Conference*, 2019.

[18] X. Yin. Initial-state detectability of stochastic discrete-event systems with probabilistic sensor failures. *Automatica*, 80:127–134, 2017.

[19] X. Yin and S. Lafortune. A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems. *IEEE Trans. Automatic Control*, 61(8):2140–2154, 2016.

[20] X. Yin and S. Lafortune. A new approach for the verification of infinite-step and k-step opacity using two-way observers. *Automatica*, 80:162–171, 2017.

[21] X. Yin and S. Lafortune. Verification complexity of a class of observational properties for modular discrete events systems. *Automatica*, 83:199–205, 2017.

[22] X. Yin and S. Lafortune. A general approach for optimizing dynamic sensor activation for discrete event systems. *Automatica*, 105:376–383, 2019.

[23] X. Yin and S. Li. Synthesis of dynamic masks for infinite-step opacity. *IEEE Trans. Automatic Control*, 2019. DOI:10.1109/TAC.2019.2916940.

[24] X. Yin, Z. Li, and W. Wang. Trajectory detectability of discrete-event systems. *Syst. Control Let.*, 119:101–107, 2018.

[25] X. Yin, Z. Li, W. Wang, and S. Li. Infinite-step opacity and k-step opacity of stochastic discrete-event systems. *Automatica*, 99:266–274, 2019.

[26] K. Zhang. The problem of determining the weak (periodic) detectability of discrete event systems is PSPACE-complete. *Automatica*, 81:217–220, 2017.

[27] K. Zhang and A. Giua. Weak (approximate) detectability of labeled petri net systems with inhibitor arcs. *IFAC-PapersOnLine*, 51(7):167–171, 2018.

[28] P. Zhao, S. Shu, F. Lin, and B. Zhang. Detectability measure for state estimation of discrete event systems. *IEEE Transactions on Automatic Control*, 64(1):433–439, 2019.

[29] L. Zhou, S. Shu, and F. Lin. N-(k_1,k_2)-detectability of discrete event systems under nondeterministic observations. In *American Control Conference*, pages 294–299, 2018.