2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Supervisory Control under Delayed Observations of Events and States

Chaohui Gong, Wenbiao Zhang, Weilin Wang, Yanwei Zang, Yacheng Pan, Lachlan L.H. Andrew, Jiayuan Liang, Hanran Zhang, Yang Xu

Abstract— In discrete event systems, it is often convenient and possible to observe directly whether or not the system is in a subset of the state space, typically after some delay, even if some event occurrences leading to the current state were not observed. We model supervisory control with the delayed observations of events and states and investigate the existence of a supervisor to obtain a given desired language accordingly. An existence verifier with polynomial run time is presented.

I. INTRODUCTION

Ramadge and Wonham [1] initiated the framework of supervisory control. Soon after that, Lin and Wonham [2] incorporate partial observation into discrete event system (DES) models and define observability to distinguish whether or not the supervisor always has sufficient observation for achieving the control goal. This paper considers supervisory control with the supervisor observing both events and states, with its observations possibly having some delay.

The original version of partial observation of the supervisory control assumes that a particular supervisor observes either all occurrences of an event or none of that event [2], [3], [4]. However, the ability of a supervisor to observe different occurrences of a particular event usually changes with the situation. For example, the orientation of a security camera can affect which events can be observed. Recently, a more fluent observation modality was introduced. With dynamic communication, agents decide which event occurrences to communicate and to whom [5]. With dynamic information acquisition, agents adjust the selection of events to be observed [6], [7]. Both dynamic communication and dynamic information acquisition are forms of dynamic observation, in which the observation of a particular event occurrence depends on the string of events leading to that occurrence [8], [9].

Delayed observations have also been studied, but exclusively for the process of a supervisor observing only events. In the early work by Balemi [10] and Balemi and Brunner [11], the system is modeled as the connection of two input/output processes, the plant and the supervisor, and each one outputs events in response to the inputs from the other. Each process immediately observes outputs by itself, but the observations of output events of the other could be delayed. Using the paradigm of supervisory control by Ramadge and Wonham [1], recent research, e.g., [12], [13], [14], focuses on the existence and construction of a supervisor that tolerates the delays in event observations. In contrast, Tripakis [15] and Zhang et al. [16] consider the delay of supervisors communicating their observations. Most of the above work measures the delay in terms of the number of events that occur in between the time an event occurs and the time a supervisor observes that event.

Moreover, for example in [17], supervisors can directly sense states along with observing events. Directly observing states is essential in important applications including robotics and medical diagnosis [17]. Kumar et al. [18] define the problem of supervisory control under incomplete state observation as “observability of predicates”. A “predicate” is a Boolean function that specifies a subset of the state space. Then, Cao et al. [19] show the reduction from the problem of state observation to the problem of event observation in supervisory control. Ushio and Takai [20] model state observation as the supervisor observing strings of state-dependent nondeterministic output symbols. All the above work implicitly assumes that state observations can always be performed instantaneously.

We introduce the combined observation of states and event occurrences with delay for supervisory control. We also develop a polynomial time verifier for testing the existence of a supervisor under the new definition of observation. Due to space limitations, all proofs have been omitted; they are available from the authors.

II. System Model and Observation Models

In this section, we discuss distributed supervisory control, introduce the transition-based event observation, and incorporate partial state-change perception with such observation.

Notation 1: The following notation is used throughout.

a. Let a suffix ! after an expression denote “is well defined”.
b. Let “:=” denote “is defined as”.
c. Let “w.r.t.”, “s.t.”, and “o.w.” denote “with respect to”,“such that”, and “otherwise”, respectively.

*This work is supported in part by PowerChina Grants KY2020-JT-12-01-2022 and HDY-CGHT20-20231262Y.

C. Gong, W. Wang (the corresponding author), J. Liang, and H. Zhang are with University of Shanghai for Science and Technology, China chgong@usst.edu.cn; wlwang@usst.edu.cn; liangjiayuan@yeah.net; hanran.c@gmail.com

W. Zhang, Y. Zang, Y. Pan, and Y. Xu are with Huadong Engineering Corporation Limited, China zhang_wb@hdec.com; zang_yw@ecidi.com;pan_yc@hdec.com; xu_y9@hdec.com

L. Andrew is with University of Melbourne, Australia llandrew@unimelb.edu.au

d. Let $[n] = \{1, 2, \ldots, n\}$ for $n \in \mathbb{N}$.

A. General Settings

We consider a DES modeled as a deterministic finite-state automaton (FSA) $G := (X, E, f, x_0)$, where X is a finite set of states, E is a finite set of events, partial function $f : X \times E \to X$ is a transition function, and x_0 is the initial state.

Let ε denote the empty event string and E^* denote the Kleene-closure of E. Given $s \in E^*$, $|s|$ is its length. Let $\overline{s} = \{u \in E^* : (\exists\, v \in E^*)\, s = uv\}$ denote the prefix-closure of a string $s \in E^*$. Given language L defined on E, let $\overline{L} = \{t \in E^* : (\exists s \in L) t \in \overline{s}\}$ be the prefix closure of L. We say L is prefix-closed if $L = \overline{L}$.

Let $T(G) := \{(x, e) \in X \times E : f(x, e)!\}$ denote the set of transitions of G. The transition function f is extended for the strings of events as follows. Given $x \in X$, $s \in E^*$, and $e \in E$, $f(x, se) := f(f(x, s), e)$ if both $f(x, s)$ and $f(f(x, s), e)$ are defined and is undefined otherwise. Let $\mathscr{L}(G) := \{s \in E^* : f(x_0, s)!\}$ denote the language generated by G. $\mathscr{L}(G)$ is prefix-closed by construction. For simplicity's sake, given $s \in \mathscr{L}(G)$, let $f(s)$ denote $f(x_0, s)$. We assume that G is accessible, i.e., $\forall x \in X[\exists s \in \mathscr{L}(G) x = f(s)]$.

Given automaton M, let $\omega(M)$ denote the state space of M. For example, $X = \omega(G)$. Given automata A and B, let $A \times B$ denote the product of A and B.

If automaton $\mathsf{G} := (X, E, f, x_0)$ is nondeterministic, its transition function has the form $f : X \times E \to 2^X$. f is extended to strings in E^* as, given $se \in E^*$ with some $e \in E$, $f(x_0, se) = \{x \in X : (\exists y \in f(x_0, s)) x \in f(y, e)\}$. Correspondingly, $\mathscr{L}(\mathsf{G}) = \{s \in E^* : f(x_0, s) \neq \emptyset\}$. A nondeterministic automaton G is called accessible if every state $x \in X$ has $x \in f(x_0, s)$ for some $s \in \mathscr{L}(\mathsf{G})$. An automaton is deterministic by default unless we say it is nondeterministic.

Binary relations $A \sqsubset A'$ and $A \sqsubseteq A'$ of automata A and A' denote "A is a subautomaton of A'" and "A is a subautomaton A' or $A = A'$", respectively.

B. Supervisory Control

Our goal is to use supervisory control to achieve a given desired language $K \subseteq \mathscr{L}(G)$. To achieve this goal, following any string in K, an event should be disabled by the supervisor if and only if an occurrence of that event causes the system to leave K. We assume K is generated by automaton $H = (Y, E, h, y_0)$, i.e., $K = \mathscr{L}(H)$. We assume $H \sqsubseteq G$.

The control is implemented by a supervisor S, in which $S : \mathscr{L}(G) \to 2^E$ is a function over the language $\mathscr{L}(G)$. With a slight abuse of notation, use S to denote both the supervisor and the function it implements. Let E_c denote the set of events controllable by S, and $E_{uc} := E \setminus E_c$ denote the set of uncontrollable events.

The controlled language was defined by Ramadge and Wonham [1], from which we inherit. Definition 1 defines the controlled language $\mathscr{L}(S/G)$ resulting from S controlling G.

Definition 1: Given G, E_{uc}, and S, the controlled language $\mathscr{L}(S/G)$ obtained by S controlling $\mathscr{L}(G)$ is iteratively defined as: Initially, $\mathscr{L}(S/G) \leftarrow \{\varepsilon\}$. Then $\mathscr{L}(S/G) \leftarrow \mathscr{L}(S/G) \cup \{se \in \mathscr{L}(G) : s \in \mathscr{L}(S/G) \wedge e \in [S(s) \cup E_{uc}]\}$ indefinitely. Equivalently,

$$\begin{aligned}\mathscr{L}(S/G) := \quad & \{t \in \mathscr{L}(G) : t = \varepsilon \\ & \vee (\forall t'e \in \overline{t}) e \in [S(t') \cup E_{uc}]\} \qquad \square\end{aligned} \tag{1}$$

Given DES G, desired language $K = \mathscr{L}(H)$, and E_c, we are interested in the existence of supervisors S under which K is the controlled language. Under perfect observation, the controllability theorem in [1] addresses such existence. From now on, we define the supervisors over the product of K and its decision space of disabling events, instead of over the product of $\mathscr{L}(G)$ and its decision space.

C. Event Observation and State Observation

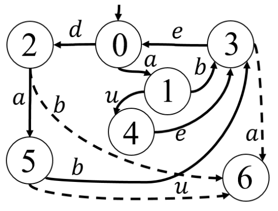
Transition-based event observation was introduced in [8], [9] and is defined as follows.

Definition 2 (information mapping): A supervisor can observe a subset of transitions in H. Function $I_\star : T(H) \to \{0, 1\}$ specifies the transitions that can be observed by S. Here $I_\star(x, e) = 1$ means that, for all $s \in \mathscr{L}(G)$ such that $x = f(s)$ (shorthand for $f(x_0, s)$), the occurrences of last event e in se is observable to S, and $I_\star(x, e) = 0$ means it is not.

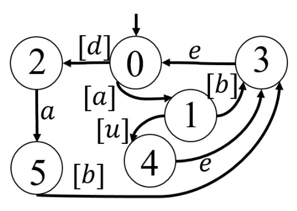
The observation of S is given by the information mapping $\theta_\star : \mathscr{L}(H) \to E^*$:

$$\theta_\star(\varepsilon) := \varepsilon;$$
$$\theta_\star(se) := \begin{cases} \theta_\star(s)e & \text{if } I_\star(h(s), e) = 1; \\ \theta_\star(s) & \text{if } I_\star(h(s), e) = 0 \end{cases}$$

for all $se \in \mathscr{L}(H)$ with some $e \in E$. $\square$



(a) System G.



(b) H and its observation by S.

Fig. 1. System G and desired system H with noted supervisor observations, in which brackets are placed around events in supervisor's diagram that cannot be observed by the supervisor.

Example 1: For the example running throughout this paper, we consider the system G in Fig. 1a that has event set $E = \{a, b, d, e, u\}$ and the supervisor with $E_c = \{a, u\}$, the sets of controllable events. Fig. 1b shows the desired system H and transition-based dynamic observations for the supervisor. In G, the transitions of broken lines are in $T(G) \setminus T(H)$, and the occurrences of events are disallowed transition-wise. $\square$

If no delay, then after s occurs, S observes $\theta_\star(s)$.

In addition to observing event occurrences, a supervisor also directly observes hints about the current state. We model this by partitioning the state space, and allowing the supervisor to observe which part contains the current state [18], [19]. We allow such observation to involve delay.

Definition 3 (partial state observation): Let Δ denote a partition of state space Y whose elements can be distinguished by supervisor S, and $\delta_0 \in \Delta$ denote the element that has initial state $y_0 \in \delta_0$. We use $\eta : Y \to \Delta$ to describe the direct observation of state space such that, for all $y \in Y$, $\eta(y)$ is the element in Δ that contains y. We assume that, to the supervisor, the order of observing states is consistent with the order of perceivable state changes. $\square$

With the obvious abuse of notation, we now define a new function $\eta : Y \times E^* \to (\Delta)^*$ to model state observations.

Definition 4: Given $H = (Y, E, h, y_0)$ and Δ, for arbitrary $y \in Y$ and arbitrary $s \in E^*$ such that $h(y,s)$ is defined, we write $s_m \in \bar{s}$ such that $|s_m| = m$. Then, for each $m \in [|s|]$, iteratively define $\eta(y, s_0) = \eta(y)$ and

$$\eta(y,s_m) := \begin{cases} \eta(y,s_{m-1}) & \text{if } \eta(h(y,s_{m-1})) = \eta(h(y,s_m)) \\ \eta(y,s_{m-1})\eta(h(y,s_m)) & \text{if } \eta(h(y,s_{m-1})) \neq \eta(h(y,s_m)) \end{cases} \square$$

Assumption 1 (state observation): This paper assumes that the observation of a state always completes before the observation of the next observed event. Moreover, this paper allows delays in observing events, but it assumes that for the supervisor S, there is a set E_b of events such that the observations of any observed event in E_b are instantaneous. Finally, the order of event observations is consistent with the order of event generations.

D. Supervisor over Observations of Events and States

We investigate the existence of a supervisor under which a given desired language can be obtained under the model of delayed partial observations specified by Definitions 2 and 4, and Assumption 1.

We need following notation to consider observations of events in the combined observation.

Notation 2: The following notation is used for specifying the supervisors considered in this paper.

a. Given H, $\theta_\star$, and E_b, for an arbitrary $s \in \mathscr{L}(H)$ and S, let a_j denote jth event in $\theta_\star(s)$. Let $s|_0 = s_0 = u_0$, and for each $k \in [l]$, $s|_k$, s_k, and u_k denote substrings of s s.t. $s|_k = s|_{k-1}a_k u_k \in \bar{s}$, $s_k = a_k u_k$, and $\theta_\star(s|_k) = \theta_\star(s|_{k-1}a_k) = \theta_\star(s|_{k-1})a_k$. Then $s = s|_l = s_0 s_1 \dots s_l = u_0 a_1 u_1 \dots a_l u_l$.
b. Let $\alpha(s) = \max\{k \in [l] : a_k \in E_b\}$ be the position of the last instantaneously observable event occurrence in $\theta_\star(s)$ if one exists, and $\alpha(s) = 0$ otherwise.

Definition 5 (pruned supervisor): Given G, H, $\theta_\star$, Δ, and E_b, a supervisor S is said to be pruned for the delayed observations specified by $(\theta_\star, \Delta, E_b)$ if for arbitrary $s = u_0 a_1 u_1 \dots a_l u_l, t = v_0 b_1 v_1 \dots b_l v_l \in \mathscr{L}(H)$ (Notation 2a), the following holds: if the decisions of S implemented following s and following t are different, i.e., $S(s) \neq S(t)$, then S can distinguish between $s|_{\alpha(s)}$ and $t|_{\alpha(t)}$ (Notation 2b) either by observing events or by observing states at the beginning or in between an observed event and another subsequently observed event, i.e., either (a) $\theta_\star(s|_{\alpha(s)}) \neq \theta_\star(t|_{\alpha(t)})$ or (b) $\alpha(s) \neq 0$ (at least one occurrence of an event in E_b is observed by S) and there exists $k \in [\alpha(s)-1] \cup \{0\}$, $\eta(h_0(s|_{k-1}), s_k) \neq \eta(h_0(t|_{k-1}), t_k)$ with $s|_{-1} = t|_{-1} = \varepsilon$.

A pruned supervisor always responds consistently, regardless of the randomness of the observations caused by delays. By definition, it satisfies conditions as follows.

a. The combined observations of events and states have the same order as that in which they occur in the system.
b. S observes any event in E_b instantaneously if it is observable, while the observations of visiting states and other events can be delayed by a random amount, but at most until an event in E_b is observed.
c. Following all strings that may possibly appear identical, depending on random delays, S uses the same decision.

We now define information mappings that combine the mode of event observation and the mode of state observation.

Definition 6 (event/state information mapping): To model the refined observation due to directly detecting states, we augment the set of events to $E = E \cup \Delta$ where an event $\delta \in \Delta$ is interpreted as entering a state in the set δ from a state outside that set. Then, define information mapping $\theta : \mathscr{L}(H) \to (E)^*$ for $i \in [A]$ as, for all $se \in \mathscr{L}(H)$ with some $e \in E$, $\theta(\varepsilon) := \delta_0$ and $\theta(se) := \theta(s)\mathsf{wz}$, in which $\mathsf{w} = e$ if $I_\star(h(s), e) = 1$ and $\mathsf{w} = \varepsilon$ if $I_\star(h(s), e) = 0$, and $\mathsf{z} = \eta(h(se))$ if $\eta(h(s)) \neq \eta(h(se))$ and $\mathsf{z} = \varepsilon$ if $\eta(h(s)) = \eta(h(se))$. $\square$

Since only an event occurrence can cause a state change, at most one partial state observation can be made per actual event occurrence. So, without delay, supervisor S observes $\theta(s)$ when a string $s \in \mathscr{L}(H)$ occurs, which means that we can write $S = S_\dagger \circ \theta$, where $S_\dagger$ is a function over $\theta(\mathscr{L}(H))$. However, the fact that state observations can be delayed after an event captures the fact that states can be observed later.

We model the delay in observations using Definition 7 and relate it to the supervisors being pruned in Definition 5 using Theorem 1 below.

Definition 7: Given $s \in \mathscr{L}(H)$, let $L(s) = \{t : \theta(s) = tt' \text{ for some } t' \in (E \setminus E_b)^*\}$. Since this depends on s only through $\theta(s)$, we can define $\psi \circ \theta(s)$ to be the string in $L(s)$ that has the minimum length among strings in $L(s)$, i.e., $[\psi \circ \theta(s) \in L(s)] \wedge [(\forall t \in L(s)) |\psi \circ \theta(s)| \leq |t|]$. $\square$

Note that, by definition, $\psi \circ \theta$ is a deterministic function.

The following theorem explains how a pruned supervisor S apply the information mapping being subjected to the delays bounded by any observation of events in E_b.

Theorem 1: Given G, H, $\theta_\star$, Δ, and E_b, a supervisor S is pruned for the delayed observations of events and states specified by $\theta_\star$, Δ, and E_b iff there exists a function $S_\dagger$ such that $S = S_\dagger \circ \psi \circ \theta$. □

Theorem 1 shows that a supervisor S making decisions over $\psi \circ \theta(\mathscr{L}(H))$ is a pruned supervisor (Definition 5).

Example 2: Let $L = \{ab, ca\}$ with $E = E_b = \{a, b, c\}$ and consider language $\bar{L}$ with its automaton model H that has initial state 0 and state space $Y = \{0, 1 = h(0,a), 2 = h(0,c), 3 = h(0,ab) = h(0,ca)\}$. Suppose supervisor S has $\Delta = \{\delta_0, \delta_1\}$ with $\delta_0 = \{0,1,2\}$ and $\delta_1 = \{3\}$. Suppose also the occurrence of a is always observable to S and any occurrence of b or c is not observable to S. Because of the delay in detecting state changes, S cannot distinguish ab from ca, which is different from the case of no delay.

III. Existence of a Supervisor

Given desired language $\mathscr{L}(H)$, we study the existence of a pruned supervisor (Definition 5) using the new information mapping (Definition 6) with the observation delay that has the upper bound of Definition 7.

A. Equivalent Condition

Theorem 2: Given $\mathscr{L}(G)$, I, Δ, E_b, E_c, and $\mathscr{L}(H)$, there exists a pruned supervisor S controlling $\mathscr{L}(G)$ such that the controlled language is $\mathscr{L}(H)$, i.e., $\mathscr{L}(S/G) = \mathscr{L}(H)$, iff for all $s \in \mathscr{L}(H)$ and all $e \in E$ such that $se \in \mathscr{L}(G) \setminus \mathscr{L}(H)$, we have (i) $e \in E_c$ and (ii) for all $t \in \mathscr{L}(H)$ with $\psi \circ \theta(t) = \psi \circ \theta(s)$, $te \notin \mathscr{L}(H)$.

Conditions (i) and (ii) above say that there is a supervisor that can disable an undesired occurrence of an event after any desired string without any observation ambiguity. We propose a constructive technique to test these conditions.

B. Extended Automaton for Observing State-Changes

Supervisor S can observe sets of states in Δ directly. To incorporate this information for supervisor S, we define automaton H' for S as follows:

Definition 8: Given $H = (Y, E, h, y_0)$, define automaton $H' = (Y', E', h', y_0')$ as follows: Let $\zeta : Y \to \zeta(Y)$ be a bijection with $Y \cap \zeta(Y) = \emptyset$ and $y_0' = \zeta(y_0)$. Then, H' has $E' = E \cup \Delta$ and transition function h specified as follows: Initially, set $h'(\delta_0) = y_0$. Iteratively, for all $e \in E$ and all $y_k, y_l \in Y$ such that $h(y_k, e) = y_l$, if $\eta(y_k) \neq \eta(y_l)$, then set $h'(y_k, e) = \zeta(y_l)$ and $h'(\zeta^i(y_l), \eta(y_l)) = y_l$; if $\eta(y_k) = \eta(y_l)$, then set $h'(y_k, e) = y_l$. Finally, Y' consists of those states in $Y \cup \zeta(Y)$ that are accessible from y_0'. □

Intuitively, H' splits a transition involving an observation of state change into two: the first represents only the event and the second represents only the state observation.

Notation 3: Given $H' = (Y', E', h', y_0')$, define $\chi : Y' \to Y$ as $\chi(y) = y$ if $y \in Y$, and $\chi(\zeta(y)) = y$ if $\zeta(y) \in Y'$. □

Based on the definition of H', we specify the observation of supervisor S by defining indicator functions I and $I_\uparrow$ and information mapping ϑ as follows:

Definition 9: Define indicator function $I : T(H') \to \{0,1\}$ as, for all $(y,e) \in T(H')$,

$$I(y,e) := \begin{cases} I_\star(y,e) & \text{if } e \in E \\ 1 & \text{if } e \in \Delta \end{cases} \qquad (2)$$

Here $I(y,e) = 1$ means that the transition (y,e) is observable to supervisor S and $I(y,e) = 0$ means that it is not. Moreover, define indicator functions $I_\uparrow : T(H) \to \{0,1\}$ as, for all $(y,e) \in T(H)$,

$$I_\uparrow(y,e) := \begin{cases} I_\star(y,e) & \text{if } e \in E_b \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

Here $I_\uparrow(y,e) = 1$ means that the transition (y,e) is always instantaneously observable to supervisor S and $I_\uparrow(y,e) = 0$ means that it is not.

Define information mapping $\vartheta : \mathscr{L}(H') \to (E')^*$ as:

$$\vartheta(\varepsilon) := \varepsilon;$$
$$\vartheta(se) := \begin{cases} \vartheta(s)e & \text{if } I(h'(s),e) = 1; \\ \vartheta(s) & \text{if } I'(h'(s),e) = 0 \end{cases}$$

for all $se \in \mathscr{L}(H')$ with some $e \in E'$. □

Notation 4: Given $s \in \mathscr{L}(H')$, let $\varphi(s) \in \bar{s}$ denote the shortest prefix of s that satisfies $\vartheta(\varphi(s)) = \vartheta(s)$. □

Definition 10: Define natural projection $P' : (E')^* \to E^*$ as follows: $P'(\varepsilon) = \varepsilon$; for all $e \in E$, $P'(e) = e$; for all $e \in \Delta$, $P'(e) = \varepsilon$; for all $st \in (E')^*$, $P'(st) = P'(s)P'(t)$. Given $L \subseteq (E')^*$, let $P'(L) = \{P'(s) : s \in L\}$ be the projection of L. Moreover, given $s \in E^*$, define $(P')^{-1}(s) = \{t \in (E')^* : P'(t) = s\}$. Given $s \in E^*$, let $\rho(s) = (P')^{-1}(s) \cap \mathscr{L}(H')$ denote the set of strings in $\mathscr{L}(H')$ having projection s. □

Lemma 1 follows immediately from the definition of ρ.

Lemma 1: For all $s \in \mathscr{L}(H)$, $|\rho(s)| \in \{1,2\}$ and, if $|\rho(s)| = 2$, we can write $\rho(s) = \{t, t\eta(h(s))\}$ for some $t \in (E')^*$. Moreover, let w be the longest string in $\rho(s)$, then, for all $se \in \mathscr{L}(H)$ with some $e \in E$, $\rho(se) = \{we, we\eta(h(se))\}$ if $\eta(h(s)) \neq \eta(h(se))$, and $\rho(se) = \{we\}$ otherwise. □

Notation 5: Given $\rho(s)$, $s \in \mathscr{L}(H)$, let $\nu(s)$ and $\mu(s)$ denote the element in $\rho(s)$ that has minimum and maximum length (t in Lemma 1), respectively. □

Lemma 2 shows the relation between θ and ϑ.

Lemma 2: For all $s \in \mathscr{L}(H)$, $\theta(s) = \vartheta(\mu(s))$. □

C. Unobservable cluster

The unobservable clusters, which are the maximal subgraphs reachable through unobservable transitions, are basic components used for the verifier.

Definition 11: Given automaton $M = (Y, E, f, y_0)$, $W \subseteq Y$, and the indicator function I of observation defined over $T(M)$, define the unobservable reach subgraph $UR(W,M,I)$ of M consisting of unobservable transitions in M that are reachable via unobservable transitions from states in $W \subseteq Y$. The transitions $T_{UR}(W,M,I)$ of $UR(W,M,I)$ are the closure of $T_{UR}(W,M,I) \leftarrow \{(y,e) \in T(M) : y \in W \wedge I(y,e) = 0\}$ under $T_{UR}(W,M,I) \leftarrow$

$T_{UR}(W,M,I) \cup \{(x,e) \in T(M) : [(\exists (x',e') \in T_{UR}(W,M,I))x = f(x',e')] \wedge [I(x,e)=0]\}$. Moreover, $UR(\{y\},M,I) \sqsubseteq M$ specifies an automaton with initial state y and set of transitions $T_{UR}(\{y\},M,I)$. □

Definition 12: Let $W = \{y \in Y' : y = y'_0 \vee [(\exists (y',e) \in T(H'))y = h'(y',e) \wedge I(y',e)=1]\}$ be the set of all states at each of which an observable transition ends. Define the unobservable cluster $c(y) \sqsubseteq H'$ for state $y \in W$ as $c(y) := UR(\{y\},H',I)$. For those $y \in Y' \setminus W$, $c(y)$ is undefined. y is called the cluster head of $c(y)$. Let Φ denote the set of clusters $c(y) \sqsubseteq H'$. □

Example 3: (Continuation of Example 1.) Given transition-based dynamic observation as in Fig. 1b, suppose $\Delta = \{\delta_0, \delta_1\}$ has $\delta_0 = \{0,2\}$, $\delta_1 = \{1,3,4,5\}$. Then, the sets of clusters for supervisors$S$ using H' are shown in Fig. 2. In the figure, for all $y \in Y$ such that $\zeta(y) \in Y'$, $\zeta(y) = y'$.

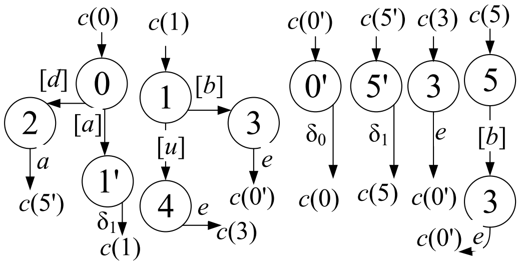


Fig. 2. Clusters for supervisor S based on H'.

Because of delay, all of the observations between two observed occurrences of events in E_b, may occur at supervisor S in a burst just before the second observed occurrence of events in E_b. The cluster $c_\uparrow(y) \sqsubseteq H$ models this.

Definition 13: Let $W = \{y \in Y : y = y_0 \vee ((\exists (y',e) \in T(H))\, y = h(y',e) \wedge I_\uparrow(y',e)=1)\}$. Define cluster $c_\uparrow(y) \sqsubseteq H$ for $y \in W$ as $c_\uparrow(y) := UR(\{y\},H,I_\uparrow)$. For $y \in Y \setminus W$, $c_\uparrow(y)$ is undefined. Let $\Phi_\uparrow$ denote the set of clusters $c_\uparrow(y) \sqsubseteq H$. □

$c_\uparrow(\cdot)$ can be obtained using the same iteration as in Definition 11 for $c(\cdot)$, except using Y, E, and $I_\uparrow$ in place of Y', E', and I.

Example 4: (Continuation of Example 3.) Given H in Fig. 1b with $E_b = E_o$, we have $I_\uparrow = I$. Then, the clusters w.r.t. $I_\uparrow$ are shown in Fig. 3.

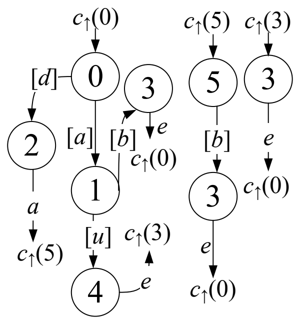


Fig. 3. Clusters of H by Definition 13.

D. Verification of Existence

To test the existence of a pruned supervisor for obtaining a given desired language, we construct a nondeterministic automaton $\mathsf{C}(G,H,H',I,E_b) := (\Omega, E \cup \Delta, g, \mathsf{c}([y'_0,y_0],y'_0))$, called the C-Machine, which is parameterized by its state space Ω and transition function $g(\cdot)$.

Each state of the C-Machine is defined by three clusters $([c(x), c_\uparrow(\chi(y))], c(z)) \in [(\Phi \cup \{c(d)\}) \times \Phi_\uparrow] \times \Phi$, where $d \notin Y'$ denotes a "dump state" used to model the delay of observation; $\chi(\cdot)$ is defined by Notation 3. $c(x)$ is any cluster the system may be in, based on the supervisor's observations. By examining any $\chi(y)$, the supervisor can find the set of events that should be enabled to ensure all desired strings are admitted. $c(z)$ is the cluster the system is actually in, and the set of events that should be disabled can be found from $c(z)$.

Let $\mathsf{c}([x,\chi(y)],z)$ denote $([c(x), c_\uparrow(\chi(y))], c(z))$ for simplicity's sake. Recall that $\omega(c(\cdot))$ is the set of states of $c(\cdot)$.

Definition 14: The transition function $g(\cdot)$ is as follows. Define, given some $\mathsf{c}(x) \in \Phi$ and some $e \in E'$, $\beta(x,e) := \{(w,e) \in T(H') : w \in \omega(\mathsf{c}(x)) \wedge I(w,e)=1\}$. Let $\mathsf{c}([x,\chi(y)],z) \in \Omega$ and $e \in E$ be arbitrary. Then, $\mathsf{c}([p,\chi(q)],r) \in g(\mathsf{c}([x,\chi(y)],z),e)$ if it satisfies the follows.

a. If $e \in E_b$, then $(\exists (u,e) \in \beta(x,e))(\exists (v,e) \in \beta(z,e))[p = q = h'(u,e) \wedge r = h'(v,e)]$.
b. If $e \in E' \setminus E_b$, then, $(\exists (v,e) \in \beta(z,e))h'(v,e) = r$ and $q = y$ and, if $\beta(x,e) \neq \emptyset$, then $(\exists (u,e) \in \beta(x,e))p = h'(u,e)$, otherwise $p = d$. □

Consider (a) in Definition 14. The system can enter any cluster such that the true head is q and the supervisor thinks it may be in any cluster headed by p, if e is observable in states u and v. Moreover, since the observation of e is instantaneous, the supervisor may update its decision according to clusters headed by some $\chi(p)$.

Consider (b) in Definition 14. We consider first e is an event, i.e., $e \in E \setminus E_b$. If both $\beta(x,e)$ and $\beta(z,e)$ are non-empty, (b) is the same as (a) except that $\chi(y)$ is not updated because the observation is not instantaneous, and so the decision must be maintained, as a different random outcome may have resulted in this e not being observed yet. If $\beta(x,e)$ is empty but $\beta(z,e)$ is non-empty, the supervisor knows that if the system is in $\mathsf{c}(x)$, the real system cannot have e being observed. But the observation of e can be delayed, the supervisor has to maintain its decision until e is observed. Consequently, the supervisor will continue to have the dumping state d as its first component until an event in E_b is observed. Then these branches of the nondeterministic evolution will terminate, but the other branches will continue. Consider then e represents a state change, $e \in \Delta$. Note that in this case e is always observable, and we can write $v = \zeta(r)$ and, if $\beta(x,e) \neq \emptyset$, $u = \zeta(p)$. Besides that, this case is the same as the case of $e \in E \setminus E_b$.

Definition 15: Define $\mathsf{C}(G,H,H',I,E_b)$ s.t. its components are the closure of the corresponding components of automaton $(\{\mathsf{c}([y'_0,y_0],y'_0)\},E',\emptyset,\mathsf{c}([y'_0,y_0],y'_0))$ under Definition 14, which specifies the expansion of its transition function $g(\cdot)$ and space of states reached by $g(\cdot)$. □

Let C denote $\mathsf{C}(G,H,H',I,E_b)$ if variables are known by the context. By definition, C is computed by starting at the initial state $\mathsf{c}([y'_0,y_0],y'_0)$ and iteratively applying Definition 14 until no transition can be augmented.

The time complexity of computing C is as follows. Ω is a subset of $[(\Phi\cup\{c(d)\})\times\Phi_{\uparrow}]\times\Phi$. Since $|Y'|\le 2|Y|$ by Definition 8, its size $|\Omega|$ has an order of $\mathscr{O}(|Y|^3)$. Implementing Definition 14 for each state in Ω has an order of $\mathscr{O}((|E|+|\Delta|)|Y|^2)$. Hence, the overall computation has an order of $\mathscr{O}((|E|+|\Delta|)|Y|^5)$. This bound is tight because the number of labeled edges in C has the same order.

Definition 16 gives a criterion for deciding the existence of a pruned supervisor.

Definition 16: Given G, H, H', I, E_b, and E_c, it is said that a state $\mathsf{c}([x,y],z)$ of C is corrupted if there exists $e\in E$ and $p\in\omega(\mathsf{c}(z))$ with $(p,e)\in T(G)\setminus T(H)$ s.t., either $e\in E_{uc}$ or there exists $q\in\omega(c_{\uparrow}(y))$ s.t. $(q,e)\in T(H)$. □

Theorem 3 relates the existence of a corrupted state in C to the existence of a supervisor for obtaining $\mathscr{L}(H)$.

Theorem 3: Given G, H, H', I, E_b, and E_c, there exists a pruned supervisor S under which the controlled language is $\mathscr{L}(H)\subseteq\mathscr{L}(G)$ exactly iff no state in C is corrupted. □

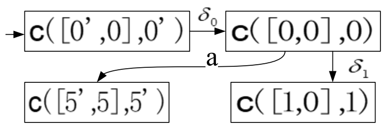


Fig. 4. $\mathsf{c}([1,0],1)$ in C is ambiguous.

Example 5: (Continuation of Example 4.) Assume that only supervisor S controls the system and the set of controllable events is $E_c=\{a,b,u\}$. In G, supervisor S must disable transitions $(2,b)$, $(3,a)$, $(5,u)$, enabling which causes the system leaving $\mathscr{L}(H)$, but all transitions in $T(H)$ should be enabled. A part of C is shown in Fig. 4. The construction starts at the initial state $\mathsf{c}([0',0],0')$ ($0'$ is a state of H'), which is not corrupt. In $\mathsf{c}([0',0],0')$, if δ_0 occurs, the system goes to $\mathsf{c}([0,0],0)$, which is not corrupt. In $\mathsf{c}([0,0],0)$, if δ_1 occurs, the system goes to $\mathsf{c}([1,0],1)$, in which a needs to be enabled at 0 in $c_{\uparrow}(0)$ but to be disabled at 3 in $c(1)$. Then, by Definition 16, $\mathsf{c}([1,0],1)$ is corrupt. Hence, by Theorem 3, no pruned supervisor can be used to obtain $\mathscr{L}(H)$. □

IV. Conclusion

Acknowledging that a supervisor is often capable of directly observing states and observation of both event and state often involves some delay, we model supervisory control with delayed dynamic event observations and state observations. We also developed a polynomial-time process for testing the existence of such a supervisor under this broadened version of partial observation.

References

[1] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," SIAM journal on control and optimization, vol. 25, no. 1, pp. 206–230, 1987.

[2] F. Lin and W. M. Wonham, "On observability of discrete-event systems," Information sciences, vol. 44, no. 3, pp. 173–198, 1988.

[3] K. Rudie and W. M. Wonham, "Think globally, act locally: decentralized supervisory control," IEEE Transactions on Automatic Control, vol. 37, no. 11, pp. 1692–1708, 1992.

[4] A. Overkamp and J. H. van Schuppen, "Maximal solutions in decentralized supervisory control," SIAM Journal on Control and Optimization, vol. 39, no. 2, pp. 492–511, 2000.

[5] W. Wang, S. Lafortune, and F. Lin, "Minimization of communication of event occurrences in acyclic discrete event systems," IEEE Transactions on Automatic Control, vol. 53, no. 9, pp. 2197–2202, 2008.

[6] W. Wang, S. Lafortune, F. Lin, and A. R. Girard, "Minimization of dynamic sensor activation in discrete event systems for the purpose of control," IEEE Transactions on Automatic Control, vol. 55, no. 11, pp. 2447–2461, 2010.

[7] W. Wang, C. Gong, and D. Wang, "Optimizing sensor activation in a language domain for fault diagnosis," IEEE Transactions on Automatic Control, vol. 64, no. 2, pp. 743–750, 2018.

[8] W. Wang, S. Lafortune, and F. Lin, "An algorithm for calculating indistinguishable states and clusters in finite-state automata with partially observable transitions," Systems & Control Letters, vol. 56, no. 9-10, pp. 656–661, 2007.

[9] W. Wang, A. R. Girard, S. Lafortune, and F. Lin, "On codiagnosability and coobservability with dynamic observations," IEEE Transactions on Automatic Control, vol. 56, no. 7, pp. 1551–1566, 2011.

[10] S. Balemi and U. Brunner, "Supervision of discrete event systems with communication delays," in 1992 American Control Conference. IEEE, 1992, pp. 2794–2798.

[11] S. Balemi, "Communication delays in connections of input/output discrete event processes," in [1992] Proceedings of the 31st IEEE Conference on Decision and Control. IEEE, 1992, pp. 3374–3379.

[12] F. Lin, "Control of networked discrete event systems: dealing with communication delays and losses," SIAM Journal on Control and Optimization, vol. 52, no. 2, pp. 1276–1298, 2014.

[13] S. Shu and F. Lin, "Deterministic networked control of discrete event systems with nondeterministic communication delays," IEEE Transactions on Automatic Control, vol. 62, no. 1, pp. 190–205, 2017.

[14] W. Wang, Y. Zang, S. Takai, L. L. Andrew, and C. Gong, "Deterministic supervisory control with flexible upper-bounds on observation delay and control delay," Automatica, vol. 144, p. 110480, 2022.

[15] S. Tripakis, "Decentralized control of discrete-event systems with bounded or unbounded delay communication," IEEE Transactions on Automatic Control, vol. 49, no. 9, pp. 1489–1501, 2004.

[16] R. Zhang, K. Cai, Y. Gan, and W. M. Wonham, "Distributed supervisory control of discrete-event systems with communication delay," Discrete Event Dynamic Systems, vol. 26, no. 2, pp. 263–293, 2016.

[17] X. C. Ding, C. Belta, and C. G. Cassandras, "Receding horizon surveillance with temporal logic specifications," in 49th IEEE Conference on Decision and Control (CDC). IEEE, 2010, pp. 256–261.

[18] R. Kumar, V. Garg, and S. I. Marcus, "Predicates and predicate transformers for supervisory control of discrete event dynamical systems," IEEE Trans. Automatic Control, vol. 38, no. 2, pp. 232–247, 1993.

[19] C. Cao, F. Lin, and Z.-H. Lin, "Why event observation: observability revisited," Discrete Event Dynamic Systems, vol. 7, no. 2, pp. 127–149, 1997.

[20] T. Ushio and S. Takai, "Nonblocking supervisory control of discrete event systems modeled by mealy automata with nondeterministic output functions," IEEE Transactions on Automatic Control, vol. 61, no. 3, pp. 799–804, 2015.