Relative Observability of Discrete-Event Systems and Its Supremal Sublanguages

Kai Cai, *Member, IEEE*, Renyuan Zhang, *Member, IEEE*, and W. M. Wonham, *Life Fellow, IEEE*

***Abstract*—We identify a new observability concept, called *relative observability*, in supervisory control of discrete-event systems under partial observation. A fixed, ambient language is given, relative to which observability is tested. Relative observability is stronger than observability, but enjoys the important property that it is preserved under set union; hence there exists the supremal relatively observable sublanguage of a given language. Relative observability is weaker than normality, and thus yields, when combined with controllability, a generally larger controlled behavior; in particular, no constraint is imposed that only observable controllable events may be disabled. We design new algorithms which compute the supremal relatively observable (and controllable) sublanguage of a given language, which is generally larger than the normal counterpart. We demonstrate the new observability concept and algorithms with a Guideway and an AGV example.**

***Index Terms*—Automata, discrete-event systems, partially-observed supervisory control, regular languages, relative observability, supremal relatively observable sublanguage.**

I. Introduction

In supervisory control of discrete-event systems, partial observation arises when the supervisor does not observe all events generated by the plant [1], [2]. This situation is depicted in Fig. 1(a), where $\mathbf{G}$ is the plant with closed behavior $L(\mathbf{G})$ and marked behavior $L_m(\mathbf{G})$, P is a natural projection that nulls unobservable events, and V^o is the supervisor under partial observation. The fundamental *observability* concept is identified in [3], [4]: observability and controllability of a language $K \subseteq L_m(\mathbf{G})$ is necessary and sufficient for the existence of a *nonblocking* supervisor V^o synthesizing K. The observability property is not, however, preserved under set union, and hence there generally does not exist the supremal observable and controllable sublanguage of a given language.

The normality concept studied in [3], [4] is stronger than observability but is algebraically well-behaved: there always exists the supremal normal and controllable sublanguage of a given language. The supremal sublanguage may be computed by methods in [5], [6]; also see a coalgebra-based method in [7]. Normality, however, imposes the constraint that controllable events cannot be disabled unless they are observable [1, Sec. 6.5]. This constraint might result in overly conservative controlled behavior.

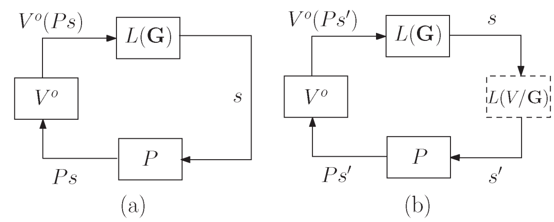


Fig. 1. Supervisory control under partial observation. $L(\mathbf{G})$ is the closed behavior of the plant, P a natural projection modeling the observation channel, V^o the supervisor under partial observation. In (b), $L(V/\mathbf{G})$ is the closed-loop controlled behavior with full observation.

To fill the gap between observability and normality, which is unsolved for more than two decades, in this paper we identify a new concept called *relative observability*. For a language $K \subseteq L_m(\mathbf{G})$, we fix an *ambient* language $\overline{C}$ such that $\overline{K} \subseteq \overline{C} \subseteq L(\mathbf{G})$ (here $\bar{\cdot}$ denotes *prefix closure*, defined in Section II). It is relative to the ambient language $\overline{C}$ that observability of K is tested. We prove that relative observability is stronger than the observability in [3], [4] (strings in $\overline{C} - \overline{K}$, if any, need to be tested), weaker than normality (unobservable controllable events may be disabled), and preserved under set union. Hence, there always exists the supremal relatively observable (and controllable) sublanguage of a given language, which is generally larger than the supremal normal counterpart, and may be synthesized by a nonblocking supervisor. This result is useful in practical situations where there may not be enough sensors available for all controllable events, or it might be too costly to provide them all.

We then design new algorithms to compute the supremal sublanguage, capable of keeping track of the ambient language. These results are demonstrated with a Guideway and an AGV example, providing quantitative evidence of improvements by relative observability as compared to normality. Note that in the special case $\overline{C} = \overline{K}$, relative observability coincides with observability for the given K. The difference, however, is that when a family of languages is considered, the ambient $\overline{C}$ in relative observability is held fixed. It is this feature that renders relative observability algebraically well-behaved.

Another special case is when the ambient $\overline{C} = L(\mathbf{G})$. As suggested by Fig. 1(a), $L(\mathbf{G})$ is a natural choice for the ambient

Manuscript received June 14, 2013; revised November 17, 2013 and March 26, 2014; accepted July 18, 2014. Date of publication July 22, 2014; date of current version February 19, 2015. This work was supported in part by the Program to Disseminate Tenure Tracking System, MEXT, Japan, the Fundamental Research Funds for the Central Universities, China, under Grant 3102014JCQ01069, and the Natural Sciences and Engineering Research Council, Canada, under Grant 7399. Recommended by Associate Editor D. Hristu-Varsakelis.

K. Cai is with the Urban Research Plaza, Osaka City University, Osaka 558-8585, Japan (e-mail: kai.cai@info.eng.osaka-cu.ac.jp).

R. Zhang is with the Department of Traffic and Control Engineering, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: ryzhang@nwpu.edu.cn).

W. M. Wonham is with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada (e-mail: wonham@control.utoronto.ca).

Digital Object Identifier 10.1109/TAC.2014.2341891

language because strings in $L(\mathbf{G})$ are observed through the channel P, but when control is in place, a more reasonable choice for the ambient $\overline{C}$ is $L(V/\mathbf{G})$, the optimal nonblocking controlled behavior under full observation, since any string in $L(\mathbf{G}) - L(V/\mathbf{G})$ is effectively prohibited by control; see Fig. 1(b). With $\overline{C} = L(V/\mathbf{G})$, the supremal relatively observable and controllable sublanguage is generally larger than the supremal normal counterpart; this is illustrated by empirical studies on a Guideway and an AGV example in Section V.

In [8], Takai and Ushio reported an observability property, formulated in a state-based form, which is preserved under a union operation of "strict subautomata." This operation does not correspond to language union. It was shown that the (marked) language of "the supremal subautomaton" with the proposed observability is generally larger than the supremal normal counterpart. As will be illustrated by examples, neither their observability property nor our relative observability generally implies the other. In the Guideway example in Section V-A, however, we present a case where our algorithm computes a strictly larger controlled behavior.

We note that, for prefix-closed languages, several procedures are developed to compute a maximal observable and controllable sublanguage, e.g., [9]–[13]. Those procedures are not, however, applicable to non-closed languages, because the resulting supervisor may be blocking. The observability concept has been extended to coobservability in decentralized supervisory control (e.g., [14], [15]), state-based observability (e.g., [16], [17]), timed observability in real-time discrete-event systems (e.g., [18], [19]), and optimal supervisory control with costs [20]. Observability and normality have also been used in modular, decentralized, and coordination control architectures (e.g., [21]–[23]). In the present paper, we focus on centralized, monolithic supervision for untimed systems in the Ramadge–Wonham language framework [1], [24], and leave those extensions of relative observability for future research.

The rest of this paper is organized as follows. Section II introduces the relative observability concept, and establishes its properties. Section III presents an algorithm to compute the supremal relatively observable sublanguage of a given language, while Section IV combines relative observability and controllability to generate controlled behavior generally larger than the normality counterpart. Section V demonstrates the results with a Guideway and an AGV example. Finally, Section VI states our conclusions.

II. Relative Observability

The plant to be controlled is modeled by a generator

$$\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m) \tag{1}$$

where Q is the finite state set; $q_0 \in Q$ is the initial state; $Q_m \subseteq Q$ is the subset of marker states; Σ is the finite event set; $\delta : Q \times \Sigma \to Q$ is the (partial) state transition function. In the usual way, δ is extended to $\delta : Q \times \Sigma^* \to Q$, and we write $\delta(q, s)!$ to mean that $\delta(q, s)$ is defined. The *closed behavior* of $\mathbf{G}$ is the language

$$L(\mathbf{G}) := \{s \in \Sigma^* | \delta(q_0, s)!\} \subseteq \Sigma^* \tag{2}$$

the *marked behavior* is

$$L_m(\mathbf{G}) := \{s \in L(\mathbf{G}) | \delta(q_0, s) \in Q_m\} \subseteq L(\mathbf{G}). \tag{3}$$

A string s_1 is a *prefix* of a string s, written $s_1 \leq s$, if there exists s_2 such that $s_1 s_2 = s$. The *(prefix) closure* of $L_m(\mathbf{G})$ is $\overline{L_m(\mathbf{G})} := \{s_1 \in \Sigma^* | (\exists s \in L_m(\mathbf{G})) s_1 \leq s\}$. In this paper, we assume $\overline{L_m(\mathbf{G})} = L(\mathbf{G})$; namely $\mathbf{G}$ is *nonblocking*.

For partial observation, let the event set Σ be partitioned into Σ_o, the observable event subset, and Σ_{uo}, the unobservable subset (i.e., $\Sigma = \Sigma_o \dot\cup \Sigma_{uo}$). Bring in the *natural projection* $P : \Sigma^* \to \Sigma_o^*$ defined according to

$$\begin{aligned} P(\epsilon) &= \epsilon, \ \epsilon \text{ is the empty string;} \\ P(\sigma) &= \begin{cases} \epsilon, & \text{if } \sigma \notin \Sigma_o, \\ \sigma, & \text{if } \sigma \in \Sigma_o; \end{cases} \\ P(s\sigma) &= P(s)P(\sigma), \ s \in \Sigma^*, \sigma \in \Sigma. \end{aligned} \tag{4}$$

In the usual way, P is extended to $P : Pwr(\Sigma^*) \to Pwr(\Sigma_o^*)$, where $Pwr(\cdot)$ denotes powerset. Write $P^{-1} : Pwr(\Sigma_o^*) \to Pwr(\Sigma^*)$ for the *inverse-image function* of P. Given two languages $L_i \subseteq \Sigma_i^*$, $i = 1, 2$, their *synchronous product* is $L_1 \| L_2 := P_1^{-1} L_1 \cap P_2^{-1} L_2 \subseteq (\Sigma_1 \cup \Sigma_2)^*$, where $P_i : (\Sigma_1 \cup \Sigma_2)^* \to \Sigma_i^*$.

Observability of a language is a familiar concept [3], [4]. Now fixing a sublanguage $C \subseteq L_m(\mathbf{G})$, we introduce *relative observability* which sets $\overline{C} \subseteq L(\mathbf{G})$ to be the *ambient language* in which observability is tested.

Definition 1: Let $K \subseteq C \subseteq L_m(\mathbf{G})$. We say K is *relatively observable* with respect to $\overline{C}$, $\mathbf{G}$, and P, or simply $\overline{C}$*-observable*, if for every pair of strings $s, s' \in \Sigma^*$ that are lookalike under P, i.e., $P(s) = P(s')$, the following two conditions hold:

$$\text{(i)} \ (\forall \sigma \in \Sigma) \ \ s\sigma \in \overline{K}, s' \in \overline{C}, s'\sigma \in L(\mathbf{G}) \Rightarrow s'\sigma \in \overline{K} \tag{5}$$

$$\text{(ii)} \ s \in K, \ s' \in \overline{C} \cap L_m(\mathbf{G}) \Rightarrow s' \in K. \tag{6}$$

Note that a pair of lookalike strings (s, s') trivially satisfies (5) and (6) if either s or s' does not belong to the ambient $\overline{C}$. For a lookalike pair (s, s') both in $\overline{C}$, relative observability requires that (i) s and s' have identical one-step continuations,[1] if allowed in $L(\mathbf{G})$, with respect to membership in $\overline{K}$; and (ii) if each string is in $L_m(\mathbf{G})$ and one actually belongs to K, then so does the other. A graphical explanation of the concept is given in Fig. 2.

If $\overline{C_1} \subseteq \overline{C_2} \subseteq L(\mathbf{G})$ are two ambient languages, it follows easily from Definition 1 that $\overline{C_2}$-observability implies $\overline{C_1}$-observability. Namely, the smaller the ambient language, the weaker the relative observability. In the special case where the ambient $\overline{C} = \overline{K}$, Definition 1 becomes the standard observability [3], [4] for the given K. This immediately implies

Proposition 1: If $K \subseteq C$ is $\overline{C}$-observable, then K is also observable.

[1] Here we consider all one-step transitions $\sigma \in \Sigma$ because we wish to separate the issue of observation from that of control. If and when control is present, as we will discuss below in Section IV, then we need to consider only controllable transitions in (5) inasmuch as the controllability requirement prevents uncontrollable events from violating (5).

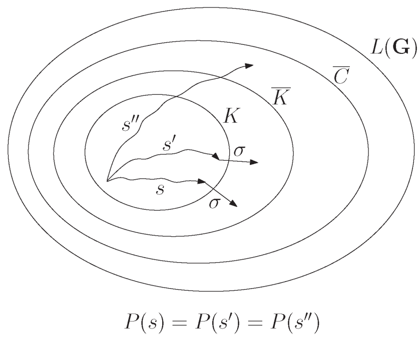


Fig. 2. Relative observability of K requires checking conditions (5) and (6) for all three lookalike strings s, s', s'' in the ambient language $\overline{C}$. For K to be $\overline{C}$-observable, condition (5) requires $s''\sigma \notin L(\mathbf{G})$, and condition (6) requires $s'' \notin L_m(\mathbf{G})$. Note that the standard observability of K [3], [4] requires checking only s, s' in $\overline{K}$.

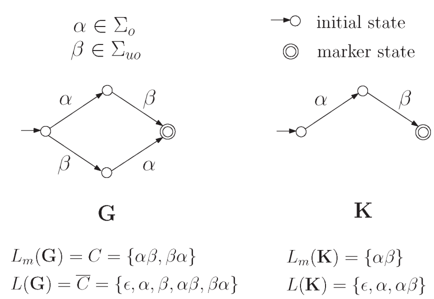


Fig. 3. $L_m(\mathbf{K})$ is observable but not relatively observable. In $L(\mathbf{K})$ the only lookalike string pair is $(\alpha, \alpha\beta)$; it is easily verified that $L_m(\mathbf{K})$ is observable. To see that $L_m(\mathbf{K})$ is not $\overline{C}$-observable, let $s = \epsilon$ and $s' = \beta$ ($\notin L(\mathbf{K})$). We have $s\alpha \in L(\mathbf{K})$, $s'\alpha \in \overline{C} = L(\mathbf{G})$, but $s'\alpha \notin L(\mathbf{K})$. This violates (5). Also consider $s = \alpha\beta$ and $s' = \beta\alpha$ ($\notin L_m(\mathbf{K})$). We have $s \in L_m(\mathbf{K})$, $s' \in \overline{C} \cap L_m(\mathbf{G})$, but $s' \notin L_m(\mathbf{K})$. This violates (6).

The reverse statement need not be true. An example is displayed in Fig. 3, of an observable language that is not relatively observable.

An important way in which relative observability differs from observability is the exploitation of a fixed ambient $\overline{C} \subseteq L(\mathbf{G})$. Let $K_i \subseteq C$, $i = 1, 2$. For (standard) observability of each K_i, one checks lookalike string pairs only in $\overline{K_i}$, ignoring all candidates permitted jointly by the other language: in this sense observability of K_i is "myopic." Consequently, both K_i being observable need not imply that their union $K_1 \cup K_2$ is observable, because the latter may be violated by a lookalike string pair $s_1 \in \overline{K_1}$ and $s_2 \in \overline{K_2}$. The fixed ambient language $\overline{C}$, by contrast, provides a "global reference": no matter which K_i one checks for relative observability, all lookalike string pairs in $\overline{C}$ must be considered. This more stringent requirement renders relative observability algebraically well-behaved, as we will see below in Section II-B. First, we shall show the relationship between relative observability and another familiar concept, *normality* [3], [4].

A. Relative Observability is Weaker than Normality

In this subsection, we show that relative observability is weaker than *normality*, a property that is also preserved by set unions [3], [4]. A sublanguage $K \subseteq C$ is $(L_m(\mathbf{G}), P)$-*normal* if

$$K = P^{-1}PK \cap L_m(\mathbf{G}). \tag{7}$$

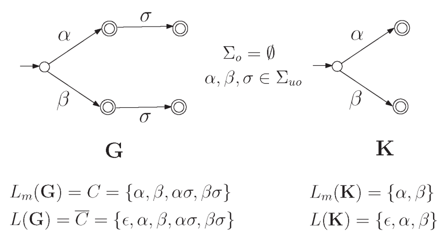


Fig. 4. $L_m(\mathbf{K})$ is relatively observable but not normal. In $L(\mathbf{K})$ all three strings are lookalike; it is easily verified that $L_m(\mathbf{K})$ is $\overline{C}$-observable. To see that $L_m(\mathbf{K})$ is not $(L_m(\mathbf{G}), P)$-normal, calculate $P^{-1}PL_m(\mathbf{K}) = P^{-1}(\epsilon) = \Sigma^*$. Thus, $P^{-1}PL_m(\mathbf{K}) \cap L_m(\mathbf{G}) = L_m(\mathbf{G}) \supsetneqq L_m(\mathbf{K})$. A similar calculation yields that $L(\mathbf{K})$ is not $(L(\mathbf{G}), P)$-normal.

If, in addition, $\overline{K}$ is $(L(\mathbf{G}), P)$-normal, then no string in $\overline{K}$ may exit $\overline{K}$ via an unobservable transition [1, Sec. 6.5]. This means, when control is present, that one cannot disable any unobservable, controllable events. Relative observability, by contrast, does not impose this restriction, i.e., one may exercise control over unobservable events.

Proposition 2: If $K \subseteq C$ is $(L_m(\mathbf{G}), P)$-normal and $\overline{K}$ is $(L(\mathbf{G}), P)$-normal, then K is $\overline{C}$-observable.

Proof: Let $s, s' \in \Sigma^*$ and $Ps = Ps'$. We must show that both (5) and (6) hold for K.

For (5), let $\sigma \in \Sigma$, $s\sigma \in \overline{K}$, $s' \in \overline{C}$, and $s'\sigma \in L(\mathbf{G})$; it will be shown that $s'\sigma \in \overline{K}$. From $s\sigma \in \overline{K}$ we have

$$P(s\sigma) \in P\overline{K} \Rightarrow P(s')P(\sigma) \in P\overline{K}$$
$$\Rightarrow s'\sigma \in P^{-1}P\overline{K}$$

Hence, $s'\sigma \in P^{-1}P\overline{K} \cap L(\mathbf{G}) = \overline{K}$ by normality of $\overline{K}$.

For (6), let $s \in K$, $s' \in \overline{C} \cap L_m(\mathbf{G})$; we will prove $s' \in K$. That $s \in K$ implies $Ps \in PK$; thus, $Ps' \in PK$, i.e., $s' \in P^{-1}PK$. Therefore, $s' \in P^{-1}PK \cap L_m(\mathbf{G}) = K$ by normality of K. □

In the proof we note that $\overline{K}$ being $(L(\mathbf{G}), P)$-normal implies condition (i) of relative observability, and independently K being $(L_m(\mathbf{G}), P)$-normal implies condition (ii). The reverse statement of Proposition 2 need not be true; an example is displayed in Fig. 4.

In Section V, we will see examples where the supremal relatively observable controlled behavior is strictly larger than the supremal normal counterpart. This is due exactly to the distinction as to whether or not one may disable controllable events that are unobservable.

We note that [8] reported an observability property which is also weaker than normality. The observability condition in [8] is formulated in a generator form, which is preserved under a particularly-defined union operation of "strict subautomata" that does not correspond to language/set union. The observability condition in [8] requires checking all state pairs (q, q') reached by lookalike strings in the whole state set Q of $\mathbf{G}$; this corresponds to checking all lookalike string pairs in $L(\mathbf{G})$. In this sense, our relative observability is weaker with the ambient language $\overline{C} \subseteq L(\mathbf{G})$ [see one example in Fig. 5(a)]. This point is also illustrated, when combined with controllability, in the Guideway example in Section V-A. However, the reverse case is also possible, as displayed in Fig. 5(b).

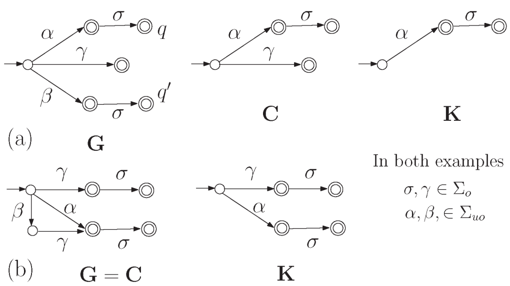


Fig. 5. Comparison with [8]. In (a), $L_m(\mathbf{K})$ is $L(\mathbf{C})$-observable; but $\mathbf{K}$ is not observable in the sense of [8], because state pair (q, q') with q in $\mathbf{K}$ and q' not in $\mathbf{K}$ violates the observability condition in [8]. In (b), $\mathbf{K}$ is observable in the sense of [8]; but $L_m(\mathbf{K})$ is not $L(\mathbf{C})$-observable, because $\gamma\sigma \in L(\mathbf{K})$, $\beta\gamma \in L(\mathbf{C})$, $\beta\gamma\sigma \in L(\mathbf{G})$, $P(\gamma) = P(\beta\gamma)$, but $\beta\gamma\sigma \notin L(\mathbf{K})$.

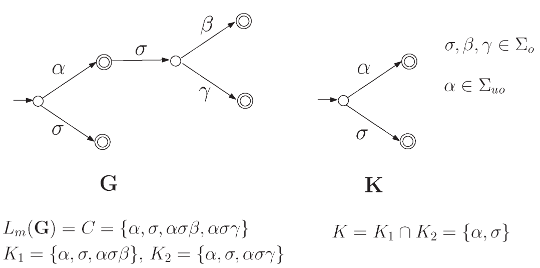


Fig. 6. Intersection of two relatively observable languages is not relatively observable. It is easily verified that both K_1 and K_2 are $\overline{C}$-observable. Their intersection K, however, is not: let $s = \epsilon$ and $s' = \alpha$; then $Ps = Ps'$, $s\sigma \in \overline{K}$, $s' \in \overline{C}$, $s'\sigma \in L(\mathbf{G})$, but $s'\sigma \notin \overline{K}$. Thus, condition (5) of relative observability is violated.

B. Supremal Relatively Observable Sublanguage

First, an arbitrary union of relatively observable languages is again relatively observable.

Proposition 3: Let $K_i \subseteq C$, $i \in I$ (some index set), be $\overline{C}$-observable. Then $K = \bigcup\{K_i | i \in I\}$ is also $\overline{C}$-observable.

Proof: Let $s, s' \in \Sigma^*$ and $Ps = Ps'$. We must show that both (5) and (6) hold for K.

For (5), let $\sigma \in \Sigma$, $s\sigma \in \overline{K}$, $s' \in \overline{C}$, and $s'\sigma \in L(\mathbf{G})$; it will be shown that $s'\sigma \in \overline{K}$. Since $\overline{K} = \overline{\bigcup K_i} = \bigcup \overline{K_i}$, there exists $j \in I$ such that $s\sigma \in \overline{K_j}$, but K_j is $\overline{C}$-observable, which yields $s'\sigma \in \overline{K_j}$. Hence, $s'\sigma \in \bigcup \overline{K_i} = \overline{K}$.

For (6), let $s \in K$, $s' \in \overline{C} \cap L_m(\mathbf{G})$; we will prove $s' \in K$. That $s \in K = \bigcup K_i$ implies that there exists $j \in I$ such that $s \in K_j$. Since K_j is $\overline{C}$-observable, we have $s' \in K_j$. Therefore, $s' \in \bigcup K_i = K$. □

While relative observability is closed under arbitrary unions, it is generally not closed under intersections. Fig. 6 provides an example for which the intersection of two $\overline{C}$-observable sublanguages is *not* $\overline{C}$-observable.

Whether or not $K \subseteq C$ is $\overline{C}$-observable, write

$$\mathcal{O}(K, C) := \left\{K' \subseteq K | K' \text{ is } \overline{C}\text{-observable}\right\} \tag{8}$$

for the family of $\overline{C}$-observable sublanguages of K. Then $\mathcal{O}(K, C)$ is an upper semilattice of sublanguages of K, with respect to the partial order $(\subseteq)$.[2] Note that the empty language $\emptyset$ is trivially $\overline{C}$-observable; thus, a member of $\mathcal{O}(K, C)$. By Proposition 3 we derive that $\mathcal{O}(K, C)$ has a unique supremal element $\sup \mathcal{O}(K, C)$ given by

$$\sup \mathcal{O}(K, C) := \bigcup \{K' | K' \in \mathcal{O}(K, C)\}. \tag{9}$$

This is the supremal $\overline{C}$-observable sublanguage of K. We state these important facts about $\mathcal{O}(K, C)$ in the following.

Theorem 1: Let $K \subseteq C$. The set $\mathcal{O}(K, C)$ is nonempty, and contains its supremal element $\sup \mathcal{O}(K, C)$ in (9).

For (9), of special interest is when the ambient language is set to equal $\overline{K}$:

$$\sup \mathcal{O}(K) := \bigcup \{K' | K' \in \mathcal{O}(K)\} \tag{10}$$

where $\mathcal{O}(K) := \{K' \subseteq K | K' \text{ is } \overline{K}\text{-observable}\}$.

Proposition 4: For $K \subseteq C \subseteq L_m(\mathbf{G})$, it holds that $\sup \mathcal{O}(K, C) \subseteq \sup \mathcal{O}(K)$.

Proof: For each $K' \subseteq K$, it follows from Definition 1 that if K' is $\overline{C}$-observable, then K' is also $\overline{K}$-observable. Hence, $\mathcal{O}(K, C) \subseteq \mathcal{O}(K)$, and $\sup \mathcal{O}(K, C) \subseteq \sup \mathcal{O}(K)$. □

Proposition 4 shows that $\sup \mathcal{O}(K)$ is the largest relatively observable sublanguage of K, for all possible choices of the ambient language. It is therefore of particular interest to characterize and compute $\sup \mathcal{O}(K)$. We do so in the next section using a generator-based approach.

III. Generator-Based Computation of $\sup \mathcal{O}(K)$

In this section, we design an algorithm to compute the supremal relatively observable sublanguage $\sup \mathcal{O}(K)$ in (10) of a given language K. This algorithm features two new mechanisms that distinguish it from those computing the supremal normal sublanguage (e.g., [5]–[7]). First, compared to [5]–[7], the algorithm has a more "fine-grained" procedure (stated precisely below) for processing *transitions* of the generators involved, because with relative observability generally fewer transitions need to be removed. Second, the algorithm keeps track of strings in the ambient language $\overline{K}$, as required by the relative observability conditions; by contrast, this is simply not an issue in [5]–[7] for the normality computation.

A. Setting

Consider a nonblocking generator $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ as in (1) with regular languages $L_m(\mathbf{G})$ and $L(\mathbf{G})$, and a natural projection $P : \Sigma^* \to \Sigma_o^*$ with $\Sigma_o \subseteq \Sigma$. Let K be an arbitrary regular sublanguage of $L_m(\mathbf{G})$. Then K can be represented by a finite-state generator $\mathbf{K} = (Y, \Sigma, \eta, y_0, Y_m)$; that is, $L_m(\mathbf{K}) = K$ and $L(\mathbf{K}) = \overline{K}$. For simplicity we assume that $\mathbf{K}$ is nonblocking, i.e., $\overline{L_m(\mathbf{K})} = L(\mathbf{K})$. Denote by n, m, respectively, the number of states and transitions of $\mathbf{K}$, i.e.,

$$\begin{aligned} n &:= |Y| \\ m &:= |\eta| = |\{(y, \sigma, \eta(y, \sigma)) \in Y \times \Sigma \times Y | \eta(y, \sigma)!\}|. \end{aligned} \tag{11}$$

We introduce

Assumption 1: $(\forall s, t \in L(\mathbf{K})) \eta(y_0, s) = \eta(y_0, t) \Rightarrow \delta(q_0, s) = \delta(q_0, t)$.

[2] For lattice theory refer to, e.g., [25], [1, Chap. 1].

If the given $\mathbf{K}$ does not satisfy Assumption 1, form the following *synchronous product* ([1], [2])

$$\mathbf{K}\|\mathbf{G} = (Y \times Q, \Sigma, \eta \times \delta, (y_0, q_0), Y_m \times Q_m) \tag{12}$$

where $\eta \times \delta : Y \times Q \times \Sigma \to Y \times Q$ is given by

$$(\eta \times \delta)((y,q),\sigma) = \begin{cases} (\eta(y,\sigma), \delta(q,\sigma)), & \text{if } \eta(y,\sigma)!, \delta(q,\sigma)!; \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

It is easily checked (e.g., [2, Sec. 2.3.3]) that $L(\mathbf{K}\|\mathbf{G}) = L(\mathbf{K}) \cap L(\mathbf{G}) = L(\mathbf{K})$, $L_m(\mathbf{K}\|\mathbf{G}) = L_m(\mathbf{K}) \cap L_m(\mathbf{G}) = L_m(\mathbf{K})$, and for every $s,t \in L(\mathbf{K}\|\mathbf{G})$ if $(\eta \times \delta)((y_0,q_0),s) = (\eta \times \delta)((y_0,q_0),t)$, then $\delta(q_0,s) = \delta(q_0,t)$. Namely $\mathbf{K}\|\mathbf{G}$ satisfies Assumption 1. Therefore, replacing $\mathbf{K}$ by the synchronous product $\mathbf{K}\|\mathbf{G}$ always makes Assumption 1 hold.

Now if for some $s \in L(\mathbf{K})$ a string $Ps \in PL(\mathbf{K})$ is observed, then the "uncertainty set" of states which s may reach in $\mathbf{K}$ is

$$U(s) := \{\eta(y_0, s') | s' \in L(\mathbf{K}), Ps' = Ps\} \subseteq Y. \tag{13}$$

If two strings have the same uncertainty set, then the following is true.

Lemma 1: Let $s,t \in L(\mathbf{K})$ be such that $U(s) = U(t)$. If $s' \in L(\mathbf{K})$ looks like s, i.e., $Ps' = Ps$, then there exists $t' \in L(\mathbf{K})$ such that $Pt' = Pt$ and $\eta(y_0,t') = \eta(y_0,s')$.

Proof: Since $s' \in L(\mathbf{K})$ and $Ps' = Ps$, by (13) we have $\eta(y_0,s') \in U(s)$. Then it follows from $U(s) = U(t)$ that $\eta(y_0,s') \in U(t)$, and hence there exists $t' \in L(\mathbf{K})$ such that $Pt' = Pt$ and $\eta(y_0,t') = \eta(y_0,s')$. □

We further adopt

Assumption 2:

$$(\forall s,t \in L(\mathbf{K}))\, \eta(y_0,s) = \eta(y_0,t) \Rightarrow U(s) = U(t). \tag{14}$$

Assumption 2 requires that any two strings reaching the same state of $\mathbf{K}$ must have the same uncertainty set. This requirement is equivalent to the "normal automaton" condition in [5], [8], which played a key role in their algorithms. In case the given $\mathbf{K}$ does not satisfy (14), replace $\mathbf{K}$ by $\mathbf{K}\|\mathbf{PK}$, where $\mathbf{PK}$ is a deterministic generator over Σ_o obtained by the *subset construction* (e.g., [1, Sec. 2.5]). $\mathbf{K}\|\mathbf{PK}$ always satisfies Assumption 2, and $L(\mathbf{K}\|\mathbf{PK}) = L(\mathbf{K})$, $L_m(\mathbf{K}\|\mathbf{PK}) = L_m(\mathbf{K})$ [8, App. A]; so, like Assumption 1, Assumption 2 entails no loss of generality. The cost, however, is that the state size of $\mathbf{K}\|\mathbf{PK}$ is at worst exponential in the state size of $\mathbf{K}$.

Let Assumptions 1 and 2 hold. We present an algorithm which produces a finite sequence of generators

$$(\mathbf{K} =)\mathbf{K}_0, \mathbf{K}_1, \ldots, \mathbf{K}_N \tag{15}$$

with $\mathbf{K}_i = (Y_i, \Sigma, \eta_i, y_0, Y_{m,i})$, $i \in [0,N]$, and a corresponding finite descending chain of languages

$$(L_m(\mathbf{K}) =) L_m(\mathbf{K}_0) \supseteq L_m(\mathbf{K}_1) \supseteq \cdots \supseteq L_m(\mathbf{K}_N)$$

such that $L_m(\mathbf{K}_N) = \sup \mathcal{O}(K)$ in (10), with the ambient language $\overline{K}$. Note that if K is already observable (in the standard sense), then $N = 0$.

B. Observational Consistency

Given $\mathbf{K}_i = (Y_i, \Sigma, \eta_i, y_0, Y_{m,i})$, $i \in [0,N]$, suppose $\overline{L_m(\mathbf{K}_i)} = L(\mathbf{K}_i)$, namely $\mathbf{K}_i$ is nonblocking. We need to check whether or not $L_m(\mathbf{K}_i)$ is $\overline{K}$-observable. To this end, we introduce a generator-based condition, called *observational consistency*. We proceed in two steps. First, let

$$\tilde{\mathbf{K}}_i = (\tilde{Y}_i, \Sigma, \tilde{\eta}_i, y_0, Y_{m,i}) \tag{16}$$

where $\tilde{Y}_i = Y_i \cup \{y_d\}$, the *dump state* $y_d \notin Y_i$, and $\tilde{\eta}_i$ is an extension of η_i which is fully defined on $\tilde{Y}_i \times \Sigma$, i.e.,

$$\tilde{\eta}_i(y_0,s) = \begin{cases} \eta_i(y_0,s), & \text{if } s \in L(\mathbf{K}_i); \\ y_d, & \text{if } s \in \Sigma^* - L(\mathbf{K}_i). \end{cases} \tag{17}$$

Clearly, the closed and marked languages of $\tilde{\mathbf{K}}_i$ satisfy $L(\tilde{\mathbf{K}}_i) = \Sigma^*$ and $L_m(\tilde{\mathbf{K}}_i) = L_m(\mathbf{K}_i)$.

Second, for each $s \in \Sigma^*$ define a set $T_i(s)$ of state pairs in $\mathbf{G}$ and $\tilde{\mathbf{K}}_i$ by

$$T_i(s) := \big\{(q,y) \in Q \times \tilde{Y}_i | (\exists s') Ps' = Ps, q = \delta(q_0,s'), y = \tilde{\eta}_i(y_0,s'), \eta(y_0,s')!\big\}. \tag{18}$$

Thus, a pair $(q,y) \in T_i(s)$ if $q \in Q$ and $y \in \tilde{Y}_i$ are reached by a common string s' that looks like s, and this s' is in $L(\mathbf{K})$, namely the ambient $\overline{K}$, because $\eta(y_0,s')!$. In (18) the last condition $\eta(y_0,s')!$ is the key to tracking strings in the ambient $\overline{K}$.

Remark 1: To compute $\sup \mathcal{O}(K,C)$ in (9), instead of $\sup \mathcal{O}(K)$ in (10), for some ambient language C satisfying $K \subseteq C \subseteq L_m(\mathbf{G})$, one should replace $T_i(s)$ in (18) by

$$T_i^C(s) := \Big\{(q,y) \in Q \times \tilde{Y}_i | (\exists s') Ps' = Ps, q = \delta(q_0,s'), y = \tilde{\eta}_i(y_0,s'), \eta^C(y_0,s')!\Big\} \tag{19}$$

where η^C is the transition function of the generator $\mathbf{C}$ with $L_m(\mathbf{C}) = C$ and $L(\mathbf{C}) = \overline{C}$. The rest follows similarly by using $T_i^C(s)$.

Definition 2: We say that $T_i(s)$ is *observationally consistent* (with respect to $\mathbf{G}$ and $\tilde{\mathbf{K}}_i$) if for all $(q,y),(q',y') \in T_i(s)$ there holds

$$(\forall \sigma \in \Sigma)\tilde{\eta}_i(y,\sigma) \neq y_d, \delta(q',\sigma)! \Rightarrow \tilde{\eta}_i(y',\sigma) \neq y_d \tag{20}$$

$$q' \in Q_m, y \in Y_{m,i} \Rightarrow y' \in Y_{m,i}. \tag{21}$$

Note that if $T_i(s)$ has only one element, then it is trivially observationally consistent. Let

$$\mathcal{T}_i := \{T_i(s) | s \in \Sigma^*, |T_i(s)| \geq 2\}. \tag{22}$$

Then $|\mathcal{T}_i| \leq 2^{|Q|\cdot(|\tilde{Y}_i|)} \leq 2^{|Q|\cdot(n+1)}$, which is finite. The following result states that checking $\overline{K}$-observability of $L_m(\mathbf{K}_i)$ is equivalent to checking observational consistency of all state pairs in each of the T_i occurring in $\mathcal{T}_i$.

Lemma 2: $L_m(\mathbf{K}_i)$ is $\overline{K}$-observable if and only if for every $T \in \mathcal{T}_i$, T is observationally consistent with respect to $\mathbf{G}$ and $\tilde{\mathbf{K}}_i$.

Proof: (If) Let $s, s' \in \Sigma^*$ and $Ps = Ps'$. We must show that both (5) and (6) hold for $L_m(\mathbf{K}_i)$.

For (5), let $\sigma \in \Sigma$, $s\sigma \in L(\mathbf{K}_i)$, $s' \in \overline{K}$, and $s'\sigma \in L(\mathbf{G})$; it will be shown that $s'\sigma \in L(\mathbf{K}_i)$. According to (18) and (17), the two state pairs $(\delta(q_0, s), \tilde{\eta}_i(y_0, s)), (\delta(q_0, s'), \tilde{\eta}_i(y_0, s'))$ belong to $T(s)$. Now $s\sigma \in L(\mathbf{K}_i)$ implies $\tilde{\eta}_i(\tilde{\eta}_i(y_0, s), \sigma) \neq y_d$ (by (17)), and $s'\sigma \in L(\mathbf{G})$ implies $\delta(\delta(q_0, s'), \sigma)!$. Since $T(s)$ is observationally consistent, by (20) we have $\tilde{\eta}_i(\tilde{\eta}_i(y_0, s'), \sigma) \neq y_d$. Then it follows from (17) that $s'\sigma \in L(\mathbf{K}_i)$.

For (6), let $s \in L_m(\mathbf{K}_i)$, $s' \in \overline{K} \cap L_m(\mathbf{G})$; we will prove $s' \in L_m(\mathbf{K}_i)$. Again $(\delta(q_0, s), \tilde{\eta}_i(y_0, s))$, $(\delta(q_0, s'), \tilde{\eta}_i(y_0, s')) \in T(s)$ according to (18) and (17). Now $s \in L_m(\mathbf{K}_i) = L_m(\tilde{\mathbf{K}}_i)$ implies $\tilde{\eta}_i(y_0, s) \in Y_{m,i}$, and $s' \in L_m(\mathbf{G})$ implies $\delta(q_0, s') \in Q_m$. Since $T(s)$ is observationally consistent, by (21) we have $\tilde{\eta}_i(y_0, s') \in Y_{m,i}$, i.e., $s' \in L_m(\tilde{\mathbf{K}}_i) = L_m(\mathbf{K}_i)$.

(Only if) Let $T \in \mathcal{T}_i$, and $(q, y), (q', y') \in T$ corresponding respectively to some s and s' with $Ps = Ps'$. We must show that both (20) and (21) hold.

For (20), let $\sigma \in \Sigma$, $\tilde{\eta}_i(y, \sigma) \neq y_d$, and $\delta(q', \sigma)!$. It will be shown that $\tilde{\eta}_i(y', \sigma) \neq y_d$. Now $(q, y) \in T$ and $\tilde{\eta}_i(y, \sigma) \neq y_d$ imply $s\sigma \in L(\mathbf{K}_i)$ (by (17)); $(q', y') \in T$ and $\delta(q', \sigma)!$ imply $s' \in \overline{K}$ and $s'\sigma \in L(\mathbf{G})$. Since $L_m(\mathbf{K}_i)$ is $\overline{K}$-observable, by (5) we have $s'\sigma \in L(\mathbf{K}_i)$, and therefore $\tilde{\eta}_i(y', \sigma) \neq y_d$.

Finally for (21), let $y \in Y_{m,i}$, $q' \in Q_m$. We will show $y' \in Y_{m,i}$. From $(q, y) \in T$ and $y \in Y_{m,i}$, $s \in L_m(\tilde{\mathbf{K}}_i) = L_m(\mathbf{K}_i)$; from $(q', y') \in T$ and $q' \in Q_m$, $s' \in \overline{K} \cap L_m(\mathbf{G})$. Since $L_m(\mathbf{K}_i)$ is $\overline{K}$-observable, by (6) we have $s' \in L_m(\mathbf{K}_i) = L_m(\tilde{\mathbf{K}}_i)$, i.e., $y' \in Y_{m,i}$. □

If there is $T \in \mathcal{T}_i$ that fails to be observationally consistent, then there exist state pairs $(q, y), (q', y') \in T$ such that either (20) or (21) or both are violated. Define two sets R_T and M_T as follows:

$$R_T := \bigcup_{\sigma \in \Sigma} \{(y, \sigma, \eta_i(y, \sigma)) \,|\, \eta_i(y, \sigma)! \;\&\; (\exists s) T = T(s) \;\&\; (q, y) \in T \;\&\; (\exists (q', y') \in T)(\delta(q', \sigma)! \;\&\; \tilde{\eta}_i(y', \sigma) = y_d)\} \tag{23}$$

$$M_T := \{y \in Y_{m,i} | (\exists s) T = T(s) \;\&\; (q, y) \in T \;\&\; (\exists (q', y') \in T)(q' \in Q_m \;\&\; y' \notin Y_{m,i})\}. \tag{24}$$

Here, $T(s)$ is as defined in (18). Thus, R_T is a collection of transitions of $\mathbf{K}_i$, each having corresponding state pairs $(q, y), (q', y') \in T$ that violate (20), while M_T is a collection of marker states of $\mathbf{K}_i$, each having corresponding state pairs that violate (21). To make T observationally consistent, all transitions in R_T have to be removed, and all states in M_T unmarked. These constitute the main steps of the algorithm below.

C. Algorithm

We present an algorithm which computes $\sup \mathcal{O}(K)$ in (10).

Algorithm 1

Input $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$, $\mathbf{K} = (Y, \Sigma, \eta, y_0, Y_m)$, and $P : \Sigma^* \to \Sigma_o^*$.

1. Set $\mathbf{K}_0 = (Y_0, \Sigma, \eta_0, y_0, Y_{m,0}) = \mathbf{K}$, namely $Y_0 = Y$, $Y_{m,0} = Y_m$, and $\eta_0 = \eta$.

2. For $i \geq 0$, calculate $\mathcal{T}_i$ as in (22) and (18) based on $\mathbf{G}$, $\mathbf{K}$, $\tilde{\mathbf{K}}_i = (\tilde{Y}_i, \Sigma, \tilde{\eta}_i, y_0, Y_{m,i})$ in (16), and P.

3. For each $T \in \mathcal{T}_i$, check if T is observationally consistent with respect to $\mathbf{G}$ and $\tilde{\mathbf{K}}_i$ (i.e., check if conditions (20) and (21) are satisfied for all $(q, y), (q', y') \in T$):

If every $T \in \mathcal{T}_i$ is observationally consistent with respect to $\mathbf{G}$ and $\tilde{\mathbf{K}}_i$, then go to Step 4 below. Otherwise, let

$$R_i := \bigcup_{T \in \mathcal{T}_i} R_T, \text{ where } R_T \text{ is defined in (23)} \tag{25}$$

$$M_i := \bigcup_{T \in \mathcal{T}_i} M_T, \text{ where } M_T \text{ is defined in (24)} \tag{26}$$

and set[3]

$$\eta_i' := \eta_i - R_i \tag{27}$$

$$Y_{m,i}' := Y_{m,i} - M_i. \tag{28}$$

Let $\mathbf{K}_{i+1} = (Y_{i+1}, \Sigma, \eta_{i+1}, y_0, Y_{m,i+1}) = \text{trim}((Y_i, \Sigma, \eta_i', y_0, Y_{m,i}'))$, where $\text{trim}(\cdot)$ removes all non-reachable and non-coreachable states and corresponding transitions of the argument generator. Now advance i to $i+1$, and go to Step 2.

4. Output $\mathbf{K}_N := \mathbf{K}_i$.

Algorithm 1 has two new mechanisms as compared to those computing the supremal normal sublanguage (e.g., [5]–[7]). First, recall that the mechanism of the normality algorithms in [5]–[7] is essentially this: If a transition σ is removed from state y of $\tilde{\mathbf{K}}_i$ reached by some string s, then remove σ from all states y' reached by a lookalike string s', i.e., $Ps = Ps'$. (In fact if σ is unobservable, then all the states y and y' as above are removed.) This (all or nothing) mechanism generally causes "overkill" of transitions (i.e., removes more transitions than necessary) in our case of relative observability, because the latter is weaker than normality and allows more permissive behavior. Indeed, some σ transitions at states y' as above may be preserved without violating relative observability. Corresponding to this feature, Algorithm 1 employs a more fine-grained mechanism: in Step 3, remove as in (27) only those transitions of $\tilde{\mathbf{K}}_i$ that violate the relative observability conditions. Moreover, the second new mechanism of Algorithm 1 is that it keeps track of strings in the ambient language $L(\mathbf{K})$ at each iteration by computing $\mathcal{T}_i$ in (22) with T_i in (18) in Step 2 above. It is these two new mechanisms that enable Algorithm 1 to compute the supremal relatively observable sublanguage $\sup \mathcal{O}(K)$ in (10).

[3]Here, η_i, η_i' denote the corresponding sets of transition triples in $Y_i \times \Sigma \times Y_i$.

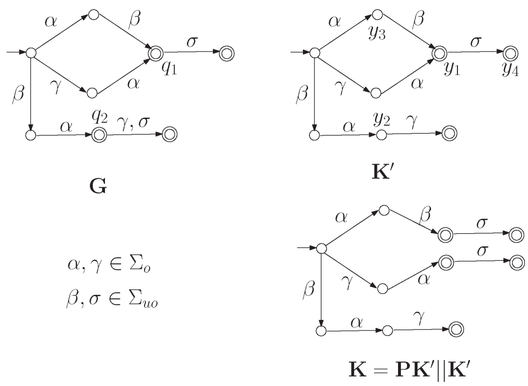


Fig. 7. Generator $\mathbf{K}'$ does not satisfy (14): strings $\alpha\beta$ and $\gamma\alpha$ both reach state y_1, but $U(\alpha\beta) = \{y_1, y_2, y_3, y_4\} \supsetneq U(\gamma\alpha) = \{y_1, y_4\}$. Now $T(\alpha)$ is not observationally consistent; indeed, two state pairs $(q_1, y_1), (q_2, y_2) \in T(\alpha)$ violate both (20) (for transition σ) and (21). Applying Algorithm 1 will remove σ at y_1 and unmark y_1, which unintentionally removes string $\gamma\alpha\sigma$ and unmarks $\gamma\alpha$. These latter two strings, however, belong to the supremal $\overline{K'}$-observable sublanguage. This undesirable situation is avoided in $\mathbf{K}$ where the strings $\alpha\beta$ and $\gamma\alpha$ are arranged to reach different states, and it is easily checked that $\mathbf{K}$ satisfies (14).

On the other hand, Algorithm 1 incurs an extra computational cost as compared to the normality algorithms in [5]–[7]. This extra cost is precisely the computation of $\mathcal{T}_i$ in (22), which in the worst case is exponential in n because $|\mathcal{T}_i| \leq 2^{(n+1)|Q|}$. While complexity is an important issue for practical computation, we shall leave for future research the problem of finding a more efficient alternative to Algorithm 1. We note in passing that if the natural projection P is an $L_m(\mathbf{G})$-*observer* (e.g., [26]), then $|\mathcal{T}_i| \leq (n+1)|Q|$ and the extra cost of Algorithm 1 is polynomial in n [27].

Algorithm 1 terminates in finite steps: in (27), the set R_T of transitions for every (observationally inconsistent) $T \in \mathcal{T}_i$ is removed; in (28), the set M_T of marker states for every (observationally inconsistent) $T \in \mathcal{T}_i$ is unmarked. At each iteration of Algorithm 1, if at Step 3 there is an observationally inconsistent T, then at least one of the two sets R_i in (25) and M_i in (26) is nonempty. Therefore at least one transition is removed and/or one marker state is unmarked. As initially in $\mathbf{K}_0 = \mathbf{K}$ there are m transitions and $|Y_m| (\leq n)$ marker states, Algorithm 1 terminates in at most $n+m$ iterations. The complexity of Algorithm 1 is $O((n+m)2^{(n+1)|Q|})$, because the search ranges $\mathcal{T}_i$ are such that $|\mathcal{T}_i| \leq 2^{(n+1)|Q|}$. Note that if $\mathbf{K}$ does *not* satisfy Assumption 2, we have to replace $\mathbf{K}$ by $\mathbf{K}\|\mathbf{PK}$ and then the complexity of Algorithm 1 is $O((2^n + m)2^{(2^n+1)|Q|})$. This double-exponential complexity is reduced to polynomial $O(n^3)$ when the natural projection P is an $L_m(\mathbf{G})$-observer [27].

Now we state our main result.

Theorem 2: Let Assumptions 1 and 2 hold. Then the output $\mathbf{K}_N$ of Algorithm 1 satisfies $L_m(\mathbf{K}_N) = \sup \mathcal{O}(K)$, the supremal $\overline{K}$-observable sublanguage of K.

Note that condition (14) of Assumption 2 on $\mathbf{K}$ is important for Algorithm 1 to generate the supremal relatively observable sublanguage, because it avoids removing and/or unmarking a string which is not intended. An illustration is given in Fig. 7.

Note also that removing a transition and/or unmarking a state may destroy observational consistency of other state pairs.

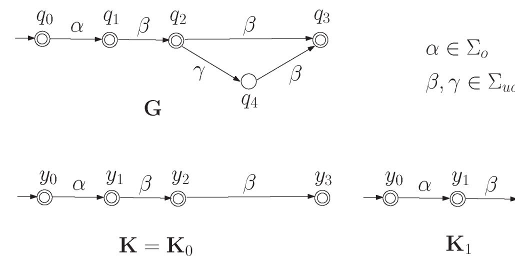


Fig. 8. In $\mathbf{K}_0$, state pairs $(q_1, y_1), (q_2, y_2) \in T(\alpha)$ are observationally consistent, while $(q_2, y_2), (q_4, y_d) \in T(\alpha)$ are not (y_d is the dump state): (20) is violated for transition β. Applying Algorithm 1 will remove β at y_2, and the result is $\mathbf{K}_1$. In $\mathbf{K}_1$, state pairs $(q_1, y_1), (q_2, y_2) \in T(\alpha)$ become observationally inconsistent: again (20) is violated for transition β. Algorithm 1 needs to be applied again to remove β at y_1.

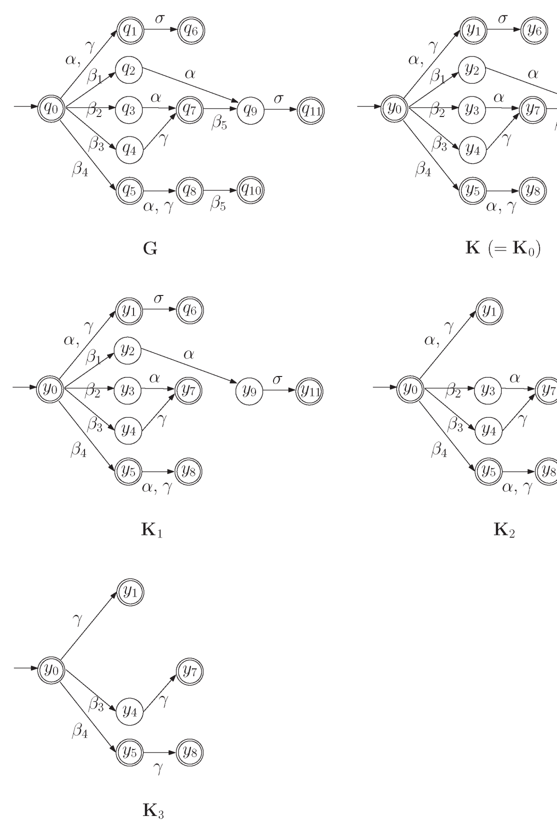


Fig. 9. Example illustration of Algorithm 1. Events $\beta_1, \ldots, \beta_5$ are unobservable and α, γ, σ observable.

Fig. 8 displays such an example. This is why all state pairs need to be checked for observational consistency at each iteration of Algorithm 1.

For just checking $\overline{C}$-observability of a given language K, with $K \subseteq C$, there is a polynomial algorithm (see [2, Sec. 3.7], [28]) for checking the standard observability which may be adapted.

The following example illustrates in detail the operations of Algorithm 1.

Example 1: Consider generators $\mathbf{G}$ and $\mathbf{K}$ displayed in Fig. 9, where events $\beta_1, \ldots, \beta_5$ are unobservable and α, γ, σ observable. These events define a natural projection P. It is easily checked that Assumption 1 holds. Also, in $\mathbf{K}$, we have $U(\alpha) = U(\gamma) = \{y_1, y_7, y_8, y_9\}$ and $U(\alpha\sigma) = U(\gamma\sigma) = \{y_6, y_{11}\}$; thus $\mathbf{K}$ satisfies (14) and Assumption 2 holds.

Apply Algorithm 1 with inputs $\mathbf{G}$, $\mathbf{K}$, and the natural projection P. Set $\mathbf{K}_0 = \mathbf{K}$, and compute $\mathcal{T}_0 = \{T_1, T_2, T_3\}$ with

$$T_1 = \{(q_0, y_0), (q_2, y_2), (q_3, y_3), (q_4, y_4), (q_5, y_5)\} \, (= T(\epsilon))$$
$$T_2 = \{(q_1, y_1), (q_7, y_7), (q_8, y_8), (q_9, y_9)\} \, (= T(\alpha) = T(\gamma))$$
$$T_3 = \{(q_6, y_6), (q_{11}, y_{11})\} \, (= T(\alpha\sigma) = T(\gamma\sigma))\,.$$

While T_1, T_3 are observationally consistent with respect to $\mathbf{K}_0$, T_2 is not; indeed, $(q_7, y_7), (q_8, y_8)$ violate (20) with event β_5. Thus, $R_0 = \{(y_7, \beta_5, y_9)\}$ and $M_0 = \emptyset$; the unobservable transition (y_7, β_5, y_9) is removed, which yields a trim generator $\mathbf{K}_1$ in Fig. 9.

The above is the first iteration of Algorithm 1. Next, compute $\mathcal{T}_1 = \{T_1, T_2, T_3, T_4, T_5\}$ with

$$T_1 = \{(q_0, y_0), (q_2, y_2), (q_3, y_3), (q_4, y_4), (q_5, y_5)\} \, (= T(\epsilon))$$
$$T_2 = \{(q_1, y_1), (q_7, y_7), (q_8, y_8), (q_9, y_d)\} \, (= T(\gamma))$$
$$T_3 = \{(q_1, y_1), (q_7, y_7), (q_8, y_8), (q_9, y_9), (q_9, y_d)\} \, (= T(\alpha))$$
$$T_4 = \{(q_6, y_d), (q_{11}, y_d)\} \, (T(\gamma\sigma))$$
$$T_5 = \{(q_6, y_d), (q_{11}, y_{11}), (q_{11}, y_d)\} \, (= T(\alpha\sigma))\,.$$

Note that $T(\alpha) \neq T(\gamma)$ and $T(\alpha\sigma) \neq T(\gamma\sigma)$ in $\mathbf{K}_1$, although $T(\alpha) = T(\gamma)$ and $T(\alpha\sigma) = T(\gamma\sigma)$ in $\mathbf{K}_0$. Now $T_2, \ldots, T_5$ are all observationally *inconsistent* with respect to $\mathbf{K}_1$, and $R_1 = \{(y_1, \sigma, y_6)\}$, $M_1 = \{y_{11}\}$. Thus, removing transition (y_1, σ, y_6), unmarking y_{11}, and trimming the result yield $\mathbf{K}_2$ in Fig. 9. This finishes the second iteration of Algorithm 1.

Compute $\mathcal{T}_3 = \{T_1, T_2\}$ with

$$T_1 = \{(q_0, y_0), (q_2, y_d), (q_3, y_3), (q_4, y_4), (q_5, y_5)\} \, (= T(\epsilon))$$
$$T_2 = \{(q_1, y_1), (q_7, y_7), (q_8, y_8), (q_9, y_d)\} \, (= T(\gamma) = T(\alpha))\,.$$

Here, T_1 is not observationally consistent, and $R_3 = \{(y_0, \alpha, y_1), (y_3, \alpha, y_7), (y_5, \alpha, y_8)\}$, $M_3 = \emptyset$. Thus removing these three transitions and trimming the result yield $\mathbf{K}_3$ in Fig. 9. This is the third iteration of Algorithm 1. Now compute $\mathcal{T}_4 = \{T_1, T_2\}$ with

$$T_1 = \{(q_0, y_0), (q_2, y_d), (q_3, y_d), (q_4, y_4), (q_5, y_5)\} \, (= T(\epsilon))$$
$$T_2 = \{(q_1, y_1), (q_7, y_7), (q_8, y_8), (q_9, y_d)\} \, (= T(\gamma))\,.$$

It is easily checked that both T_1 and T_2 are observationally consistent with respect to $\mathbf{K}_3$; by Lemma 2, $L_m(\mathbf{K}_3)$ is $L(\mathbf{K})$-observable. Hence Algorithm 1 terminates after four iterations, and returns $\mathbf{K}_3$. By Theorem 2, $L_m(\mathbf{K}_3)$ is in fact the supremal $L(\mathbf{K})$-observable sublanguage of $L(\mathbf{K})$. By contrast, the supremal normal sublanguage of $L(\mathbf{K})$ is empty.

We now prove Theorem 2. We will need this observation: from $\mathbf{K}_i$ to $\mathbf{K}_{i+1}$ in Step 3 of Algorithm 1 above, for all $s, t \in \Sigma^*$ if $\eta_{i+1}(y_0, s)!$, $\eta_{i+1}(y_0, t)!$, then $\eta_i(y_0, s)!$, $\eta_i(y_0, t)!$, and

$$\eta_{i+1}(y_0, s) = \eta_{i+1}(y_0, t) \Rightarrow \eta_i(y_0, s) = \eta_i(y_0, t). \tag{29}$$

Proof of Theorem 2: We show $L_m(\mathbf{K}_N) = \sup \mathcal{O}(K)$. First, it is guaranteed by Algorithm 1 that for the output $\mathbf{K}_N$, all the corresponding $T \in \mathcal{T}_N$ are observationally consistent; hence Lemma 2 implies that $L_m(\mathbf{K}_N)$ is $\overline{K}$-observable.

It remains to prove that if $K' \in \mathcal{O}(K)$, then $K' \subseteq L_m(\mathbf{K}_N)$. We proceed by induction on the iterations $i = 0, 1, 2, \ldots$ of Algorithm 1. Since $K' \subseteq K = L_m(\mathbf{K})$, we have $K' \subseteq L_m(\mathbf{K}_0)$. Suppose now $K' \subseteq L_m(\mathbf{K}_i)$; we show that $K' \subseteq L_m(\mathbf{K}_{i+1})$. Let $w \in K'$; by hypothesis $w \in L_m(\mathbf{K}_i)$. It will be shown that $w \in L_m(\mathbf{K}_{i+1})$ as well.

First, suppose on the contrary that $w \notin L(\mathbf{K}_{i+1})$. Since $w \in L(\mathbf{K}_i)$, there exist $t \in \Sigma^*$ and $\sigma \in \Sigma$ such that $t\sigma \leq w$, $\eta_i(y_0, t) =: y \in Y_i$, and $(y, \sigma, \eta_i(y, \sigma)) \in R_i$ in (25). Then there is $T \in \mathcal{T}_i$ such that $(y, \sigma, \eta_i(y, \sigma)) \in R_T$ in (23), and T is *not* observationally consistent ((20) is violated). Since K' is $\overline{K}$-observable and $t \in \overline{K'}$, Lemma 2 implies that $T(t)$ is observationally consistent, and thus $T(t) \neq T$.

Now let $s \in \Sigma^*$ be such that $s \neq t$, $\eta_i(y_0, s) = \eta_i(y_0, t) = y$, and $T(s) = T$. Then by (23) there exists $(q', y') \in T(s)$ such that $\delta(q', \sigma)!$ and $\tilde{\eta}_i(y', \sigma) = y_d$. Let $s' \in L(\mathbf{K}_0) = L(\mathbf{K})$ be such that $Ps = Ps'$, $\delta(q_0, s') = q'$, and $\tilde{\eta}_i(y_0, s') = y'$. Whether or not $y' = y_d$, there must exist $s'_1, u \in \Sigma^*$ such that $s'_1 u = s'$, $\tilde{\eta}_i(y_0, s'_1) \neq y_d$ (i.e., $\eta_i(y_0, s'_1)!$), and the following is true: if $u = \epsilon$ then $\tilde{\eta}_i(y_0, s'_1\sigma) = y_d$; otherwise, for each $u_1 \in \overline{\{u\}} - \{\epsilon\}$, $\tilde{\eta}_i(y_0, s'_1 u_1) = y_d$. We claim that $u \in \Sigma_{uo}^*$, i.e., an unobservable string. Otherwise, if there exist $u_1 \leq u$ and $\alpha \in \Sigma_o$ such that $u_1\alpha \leq u$, then by $Ps = Ps'$ there is $s_1 \leq s$ such that $s_1\alpha \leq s$ and $Ps_1 = P(s'_1 u_1)$. Since $\tilde{\eta}_i(y_0, s'_1 u_1 \alpha) = y_d$, we have $(\eta_j(y_0, s_1), \alpha, \eta_j(y_0, s_1\alpha)) \in R_j$ for some $j < i$. Hence, $s \notin L(\mathbf{K}_i)$, which contradicts our choice of s such that $\eta_i(y_0, s) = \eta_i(y_0, t) = y$.

Now $u \in \Sigma_{uo}^*$ and $s'_1 u = s'$ imply $Ps'_1 = Ps' = Ps$. Since $\eta_i(y_0, s) = \eta_i(y_0, t) = y$, by repeatedly using (29) we derive $\eta_0(y_0, s) = \eta_0(y_0, t) = y$. Then by Assumption 2 and Lemma 1, there exists $t' \in L(\mathbf{K}_0)$ such that $Pt = Pt'$ and $\eta_0(y_0, t') = \eta_0(y_0, s'_1)$. Thus, $\eta_0(y_0, t'u) = \eta_0(y_0, s'_1 u)$. It then follows from Assumption 1 and $\eta_0 = \eta$ that $\delta(q_0, t'u) = \delta(q_0, s'_1 u) = q'$ and $\delta(\delta(q_0, t'u), \sigma)!$. On the other hand, $\tilde{\eta}_i(\tilde{\eta}_i(y_0, t'u), \sigma) = \tilde{\eta}_i(\tilde{\eta}_i(y_0, s'_1 u), \sigma) = y_d$. Since $P(t'u) = Pt' = Pt$, we have $(\delta(q_0, t'u), \tilde{\eta}_i(y_0, t'u)) \in T(t)$. This implies that $T(t)$ is *not* observationally consistent, which contradicts that K' is $\overline{K}$-observable. Therefore, $w \in L(\mathbf{K}_{i+1})$.

Next, suppose $w \in L(\mathbf{K}_{i+1}) - L_m(\mathbf{K}_{i+1})$. Since $w \in L_m(\mathbf{K}_i)$, we have $\eta_i(y_0, w) =: y_m$ and $y_m \in M_i$ in (26). Then there is $T \in \mathcal{T}_i$ such that $y_m \in M_T$ in (24), and T is *not* observationally consistent ((21) is violated). Since K' is $\overline{K}$-observable and $w \in K'$, Lemma 2 implies that $T(w)$ is observationally consistent, and thus $T(w) \neq T$.

Now let $v \in \Sigma^*$ be such that $v \neq w$, $\eta_i(y_0, v) = \eta_i(y_0, w) = y_m$, and $T(v) = T$. Then by (24) there exists $(q'_m, y'_m) \in T(v)$ such that $q'_m \in Q_m$ and $y'_m \notin Y_{m,i}$. Let $v' \in L(\mathbf{K}_0) = L(\mathbf{K})$ be such that $Pv = Pv'$, $\delta(q_0, v') = q'_m$, and $\tilde{\eta}_i(y_0, v') = y'_m$. Whether or not $y'_m = y_d$, by a similar argument to the one above we derive that there exists w', with $Pw = Pw'$, such that $\delta(y_0, w') = \delta(y_0, v') = q'_m$, $\eta_0(y_0, w') = \eta_0(y_0, v')$, and $\tilde{\eta}_i(y_0, w') = \tilde{\eta}_i(y_0, v') = y'_m \notin Y_{m,i}$. It follows that $(\delta(q_0, w'), \tilde{\eta}_i(y_0, w')) \in T(w)$. This implies that $T(w)$ is *not* observationally consistent, which contradicts that K' is $\overline{K}$-observable. Therefore, $w \in L_m(\mathbf{K}_{i+1})$, and the proof is complete. □

IV. Supremal Relatively Observable and Controllable Sublanguage

Consider a plant $\mathbf{G}$ as in (1) with $\Sigma = \Sigma_c \dot{\cup} \Sigma_u$, where Σ_c is the controllable event subset and Σ_u the uncontrollable subset.

A language $K \subseteq L_m(\mathbf{G})$ is *controllable* (with respect to $\mathbf{G}$ and Σ_u) if $\overline{K}\Sigma_u \cap L(\mathbf{G}) \subseteq \overline{K}$. A *supervisory control* for $\mathbf{G}$ is any map $V : L(\mathbf{G}) \to \Gamma$, where $\Gamma := \{\gamma \subseteq \Sigma | \gamma \supseteq \Sigma_u\}$. Then the closed-loop system is $V/\mathbf{G}$, with closed behavior $L(V/\mathbf{G})$ and marked behavior $L_m(V/\mathbf{G})$. Let $\Sigma_o \subseteq \Sigma$ and $P : \Sigma^* \to \Sigma_o^*$ be the natural projection. We say V is *feasible* if $(\forall s, s' \in L(\mathbf{G}))\ P(s) = P(s') \Rightarrow V(s) = V(s')$, and V is *nonblocking* if $\overline{L_m(V/\mathbf{G})} = L(V/\mathbf{G})$.

It is well known [3] that a feasible nonblocking supervisory control V exists which synthesizes a nonempty sublanguage $K \subseteq L_m(\mathbf{G})$ if and only if K is both controllable and observable.[4] When K is not observable, however, there generally does not exist the supremal controllable and observable sublanguage of K. In this case, the stronger normality condition is often used instead of observability, so that one may compute the supremal controllable and normal sublanguage of K [3], [4]. With normality (K is $(L_m(\mathbf{G}), P)$-normal and $\overline{K}$ is $(L(\mathbf{G}), P)$-normal), however, no unobservable controllable event may be disabled; for some applications the resulting controlled behavior might thus be overly conservative.

This section will present an algorithm which computes, for a given language $K \subseteq L_m(\mathbf{G})$, a controllable and relatively observable sublanguage K_∞ that is generally larger than the supremal controllable and normal sublanguage of K. In particular, it allows disabling unobservable controllable events. Being relatively observable, K_∞ is also observable and controllable, and thus may be synthesized by a feasible nonblocking supervisory control.

Given a language $K \subseteq L_m(\mathbf{G})$, whether controllable or not, write $\mathcal{C}(K) := \{K' \subseteq K | K' \text{ is controllable}\}$ for the family of controllable sublanguages of K. Then $\mathcal{C}(K)$ is nonempty ($\emptyset$ belongs) and has a unique supremal element $\sup \mathcal{C}(K) := \bigcup\{K' | K' \in \mathcal{C}(K)\}$ [1]. An algorithm is proposed in [24] that computes the supremal controllable sublanguage $\sup \mathcal{C}(K)$: with input generators $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ and $\mathbf{K} = (Y, \Sigma, \eta, y_0, Y_m)$ representing $L_m(\mathbf{G})$ and K, the algorithm returns a generator $\mathbf{H}$ such that $L_m(\mathbf{H}) = \sup \mathcal{C}(K)$. Henceforth, we refer to this algorithm as Algorithm 2. By [24], Algorithm 2 terminates in at most $|Y|$ iterations and has polynomial complexity $|Q| \cdot |Y|$.

Now we design an algorithm, which iteratively applies Algorithms 1 and 2, to compute a controllable and relatively observable sublanguage of K. Let Assumptions 1 and 2 in Section III hold.

Algorithm 3

Input $\mathbf{G}$, $\mathbf{K}$, and $P : \Sigma^* \to \Sigma_o^*$.

1. Set $\mathbf{K}_0 = \mathbf{K}$.

2. For $i \geq 0$, apply Algorithm 2 with inputs $\mathbf{G}$ and $\mathbf{K}_i$. Compute $\mathbf{H}_i$ such that $L_m(\mathbf{H}_i) = \sup \mathcal{C}(L_m(\mathbf{K}_i))$.

3. Apply Algorithm 1 with inputs $\mathbf{G}$, $\mathbf{H}_i$, and $P : \Sigma^* \to \Sigma_o^*$. Obtain $\mathbf{K}_{i+1}$ such that $L_m(\mathbf{K}_{i+1}) = \sup \mathcal{O}(L_m(\mathbf{H}_i)) = \sup \mathcal{O}(\sup \mathcal{C}(L_m(\mathbf{K}_i)))$. If $\mathbf{K}_{i+1} = \mathbf{K}_i$, then output $\mathbf{K}_\infty = \mathbf{K}_{i+1}$. Otherwise, advance i to $i+1$ and go to Step 2.

[4] Here we let $L_m(V/\mathbf{G}) = L(V/\mathbf{G}) \cap K$, namely marking is part of supervisory control V's action. In this way we do not need to assume that K is $L_m(\mathbf{G})$-closed, i.e., $K = \overline{K} \cap L_m(\mathbf{G})$ [1, Sec. 6.3].

Note that in applying Algorithm 1 at Step 3, the ambient language successively shrinks to the supremal controllable sublanguage $\sup \mathcal{C}(L_m(\mathbf{K}_i))$ computed by Algorithm 2 at the immediately previous Step 2 of Algorithm 3. Thus, every $L_m(\mathbf{K}_{i+1})$ is relatively observable with respect to $\sup \mathcal{C}(L_m(\mathbf{K}_i))$. This choice of ambient languages is based on the intuition that at each iteration i, any behavior outside $\sup \mathcal{C}(L_m(\mathbf{K}_i))$ may be effectively disabled by means of control, and hence is discarded when observability is tested. The successive shrinking of ambient languages is useful in computing less restrictive controlled behavior, as compared to the algorithm in [8] which is equivalent to fixing the ambient language at $L(\mathbf{G})$. An illustration is the Guideway example in the next section.

Since Algorithms 1 and 2 both terminate in finite steps, and there can be at most $|Y|$ applications of the two algorithms, Algorithm 3 also terminates in finite steps. This means that the sequence of languages

$$L_m(\mathbf{K}_0) \supseteq L_m(\mathbf{H}_1) \supseteq L_m(\mathbf{K}_1) \supseteq L_m(\mathbf{H}_2) \supseteq L_m(\mathbf{K}_2) \supseteq \cdots$$

is finitely convergent to $L_m(\mathbf{K}_\infty)$. The complexity of Algorithm 3 is exponential in $|Y|$ because Algorithm 1 is of this complexity.

Note that in computing the supremal relatively observable sublanguages at Step 3, in particular for R_T in (23), we may restrict attention only to Σ_c because uncontrollable transitions are dealt with by the controllability requirement.

Theorem 3: $L_m(\mathbf{K}_\infty)$ is controllable and observable, and contains at least the supremal controllable and normal sublanguage of K.

Proof: For the first statement, let $\mathbf{K}_\infty = \mathbf{K}_{i+1} = \mathbf{K}_i$ for some $i \geq 0$. According to Steps 2 and 3 of Algorithm 3, the latter equality implies that $L_m(\mathbf{K}_\infty)$ is controllable and $\overline{\sup \mathcal{C}(L_m(\mathbf{K}_i))}$-observable. Therefore, $L_m(\mathbf{K}_\infty)$ is controllable and observable by Proposition 1.

For the second statement, set up a similar algorithm to Algorithm 3 but replace Step 3 by a known procedure to compute the supremal normal sublanguage ([5], [6]). Denote the resulting generators by $\mathbf{K}'_i$. Then by Proposition 2, $L_m(\mathbf{K}_i) = \sup \mathcal{O}(\sup \mathcal{C}(L_m(\mathbf{K}_{i-1}))) \supseteq L_m(\mathbf{K}'_i)$, for all $i \geq 1$. Now suppose the new algorithm terminates at the jth iteration. Then Algorithm 3 must terminate at the jth iteration or earlier, because normality implies relative observability. Therefore, $L_m(\mathbf{K}'_j) \subseteq L_m(\mathbf{K}_j)$, i.e., $L_m(\mathbf{K}_\infty)$ contains the supremal controllable and normal sublanguage of K. □

Algorithm 3 has been implemented as a procedure in [29]. To empirically demonstrate Theorem 3, the next section applies Algorithm 3 to study two examples, Guideway and AGV.

V. Examples

Our first example, Guideway, illustrates that Algorithm 3 returns an observable and controllable language that is invariably no smaller than and may actually be larger than the one based on normality. The second example, the AGV system, provides computational results to demonstrate Algorithm 3 as well as to compare relative observability with normality.

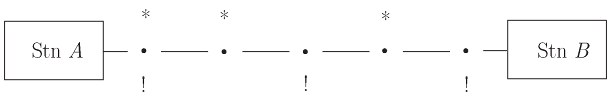


Fig. 10. Guideway: stations A and B are connected by a single one-way track from A to B. The track consists of four sections, with stoplights ($*$) and detectors (!) installed at various section junctions as displayed.

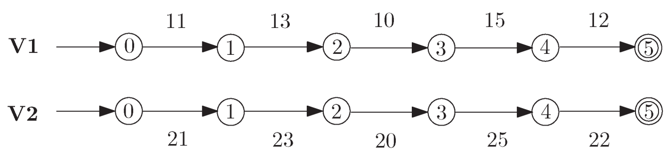


Fig. 11. Vehicle generator model.

A. Control of a Guideway Under Partial Observation

We demonstrate relative observability and Algorithm 3 with a Guideway example, adapted from [1, Sec. 6.6]. As displayed in Fig. 10, stations A and B on a Guideway are connected by a single one-way track from A to B. The track consists of four sections, with stoplights ($*$) and detectors (!) installed at various section junctions. Two vehicles, $\mathbf{V}_1$ and $\mathbf{V}_2$, use the Guideway simultaneously. Their generator models are displayed in Fig. 11; $\mathbf{V}_i$, $i = 1, 2$, is at state 0 (station A), state j (while traveling in section $j = 1, \ldots, 4$), or state 5 (station B). The plant $\mathbf{G}$ to be controlled is $\mathbf{G} = \mathbf{V}_1 \| \mathbf{V}_2$.

To prevent collision, control of the stoplights must ensure that $\mathbf{V}_1$ and $\mathbf{V}_2$ never travel on the same section of track simultaneously: i.e., ensure mutual exclusion of the state pairs (j, j), $j = 1, \ldots, 4$. Let $\mathbf{K}$ be a generator enforcing this specification. Here according to the locations of stoplights ($*$) and detectors (!) displayed in Fig. 10, we choose controllable events to be $i1, i3, i5$, and unobservable events $i3, i5$, $i = 1, 2$. The latter define a natural projection P.

First, applying Algorithm 2, with inputs $\mathbf{G}$, $\mathbf{K}$, and Σ_c, we obtain the full-observation monolithic supervisor, with 30 states, 40 transitions, and marked language $\sup \mathcal{C}(L_m(\mathbf{G} \| \mathbf{K}))$. Now applying Algorithm 3 we obtain the generator displayed in Fig. 12; Algorithm 3 terminates after just one iteration. The resulting controlled behavior is verified to be controllable and observable (as asserted by Theorem 3). Moreover, it is strictly larger than the supremal normal and controllable sublanguage represented by the generator displayed in Fig. 13. The reason is as follows. After string 11.13.10, $\mathbf{V}_1$ is at state 3 (section 3) and $\mathbf{V}_2$ at 0 (station A). With relative observability, either $\mathbf{V}_1$ executes event 15 (moving to state 4) or $\mathbf{V}_2$ executes 21 (moving to state 1); in the latter case, the controller disables event 23 after execution of 21 to ensure mutual exclusion at (3,3) because event 20 is uncontrollable. With normality, however, event 23 cannot be disabled because it is unobservable; thus 21 is disabled after string 11.13.10, and the only possibility is that $\mathbf{V}_1$ executes 15. In fact, 21 is kept disabled until the observable event 12 occurs, i.e., $\mathbf{V}_1$ arrives at station B.

For this example, the algorithm in [8] yields the same generator as the one in Fig. 13; indeed, states 12 and 13 of the generator in Fig. 12 must be removed in order to meet the observability definition in [8]. Thus, this example illustrates

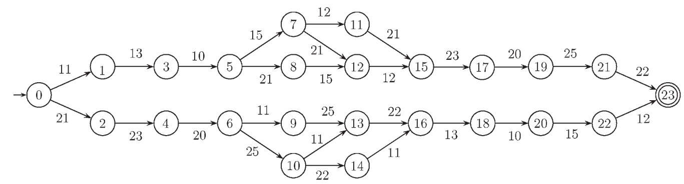


Fig. 12. Supremal relatively observable and controllable sublanguage.

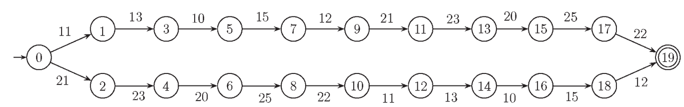


Fig. 13. Supremal normal and controllable sublanguage.

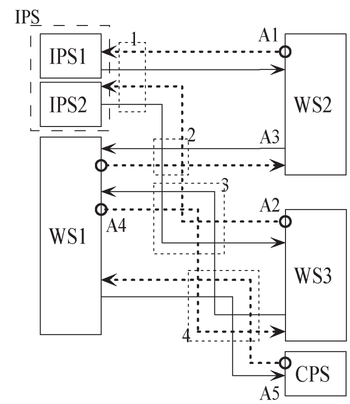


Fig. 14. AGV: system configuration.

that our algorithm can obtain a larger controlled behavior compared to [8].

B. Control of an AGV System Under Partial Observation

We now apply Algorithm 3 to study a larger example, a system of five automated guided vehicles (AGVs) serving a manufacturing workcell, in the version of [1, Sec. 4.7], originally adapted from [30].

As displayed in Fig. 14, the workcell consists of two input parts stations IPS1, IPS2 for parts of types 1 and 2, three workstations WS1, WS2, WS3, and one completed parts station CPS. Five independent AGVs—AGV1, …, AGV5—travel in fixed criss-crossing routes, loading/unloading and transporting parts in the cell. We model the synchronous product of the five AGVs as the plant to be controlled, on which three types of control specifications are imposed: the mutual exclusion (i.e., single occupancy) of shared zones (dashed squares in Fig. 14), the capacity limit of workstations, and the mutual exclusion of the shared loading area of the input stations. The generator models of plant components and specifications are displayed in Fig. 15; here odd numbered events are controllable, and there are ten such events, $i1, i3, i = 1, \ldots, 5$. For observable events, we will consider different subsets of events below. The reader is referred to [1, Sec. 4.7] for the detailed interpretation of events.

Under full observation, we obtain by Algorithm 2 the monolithic supervisor of 4406 states and 11 338 transitions. Then

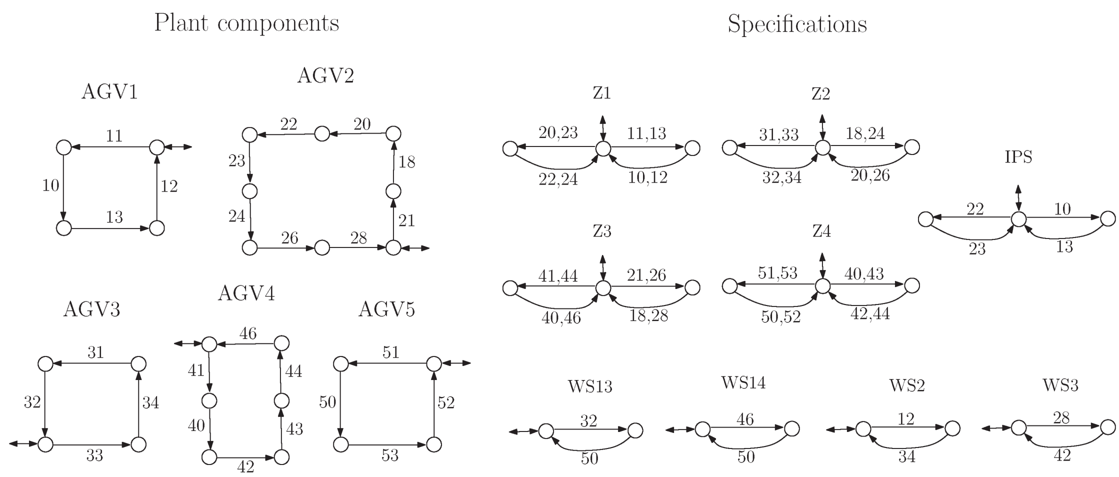


Fig. 15. Generators of plant components and specifications.

TABLE I
TEST RESULTS OF ALGORITHM 3 FOR DIFFERENT SUBSETS OF UNOBSERVABLE EVENTS IN THE AGV SYSTEM

$\Sigma_{uo} = \Sigma - \Sigma_o$	State # of rel. obs. supervisor	State # of normal supervisor	Iteration # of Alg. 3	Iteration # of Alg. 1
{13}	4406	3516	1	1
{21}	4348	0	1	399
{41,51}	3854	0	2	257
{31,43}	4215	1485	1	233
{11,31,41}	163	0	1	28
{13,23,31,33, 41,43,51,53}	579	0	3	462

we select different subsets of controllable events to be unobservable, and apply Algorithm 3 to compute the corresponding supervisors which are relatively observable and controllable. The computational results are displayed in Table I; the supervisors are state minimal, and controllability, observability, and normality are independently verified. All computations and verifications are done by procedures implemented in [29].

The cases in Table I show considerable differences in state size between relatively observable and controllable supervisors and the normal counterparts. In the case $\Sigma_{uo} = \{13\}$, the monolithic supervisor is in fact observable in the standard sense; thus, Algorithms 1 and 3 both terminate after 1 iteration, and no transition removal or state unmarking was done. Therefore the controlled behavior of the resulting relatively observable supervisor is identical to that of the full-observation supervisor. By contrast, the normal supervisor loses 890 states, and its controlled behavior can only be less permissive.

The contrast in state size is more significant in the case $\Sigma_{uo} = \{21\}$: the normal supervisor is empty (i.e., no controlled behavior is allowed at all); while the relatively observable supervisor loses merely 58 states compared to the full-observation supervisor, and hence has more permissive controlled behavior than the normal supervisor. The last row of Table I shows a case where only two out of ten controllable events, 11 and 21, are observable. Still, relative observability produces a 579-state supervisor, which has more permissive controlled behavior than the normal supervisor that is empty and allows no controlled behavior. Indeed, the normal supervisor allows no controlled behavior when only events 41 and 51 are unobservable (the third case in Table I).

Note from the state sizes of relatively observable supervisors in Table I that no state increase occurs compared to the full-observation supervisor. In addition, the last two columns of Table I suggest that Algorithm 3 with Algorithm 1 embedded terminates reasonably fast.

VI. CONCLUSION

We have identified the new concept of relative observability, and proved that it is stronger than observability, weaker than normality, and preserved under set union. Hence, there exists the supremal relatively observable sublanguage of a given language. In addition we have provided an algorithm to effectively compute the supremal sublanguage. This result thereby solved a longstanding open problem in supervisory control under partial observation.

Combined with controllability, moreover, relative observability generates generally larger controlled behavior than the normality counterpart. This has been demonstrated empirically with a Guideway example and an AGV example.

Newly identified, the algebraically well-behaved concept of relative observability may be expected to impact several closely related topics such as coobservability, decentralized supervisory control, stated-based observability, and observability of timed discrete-event systems. In future work we aim to explore these directions.

REFERENCES

[1] W. M. Wonham, "Supervisory Control of Discrete-Event Systems," Systems Control Group, Elect. and Comput. Eng. Dept., Univ. of Toronto, updated Jul. 1, 2014. [Online]. Available: http://www.control.toronto.edu/DES

[2] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. New York, NY, USA: Springer, 2007.

[3] F. Lin and W. M. Wonham, "On observability of discrete-event systems," *Inf. Sci.*, vol. 44, pp. 173–198, 1988.

[4] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya, "Supervisory control of discrete-event processes with partial observations," *IEEE Trans. Autom. Control*, vol. 33, no. 3, pp. 249–260, Mar. 1988.

[5] H. Cho and S. I. Marcus, "On supremal languages of classes of sublanguages that arise in supervisor synthesis problems with partial observation," *Math. Control, Signals, Syst.*, vol. 2, no. 1, pp. 47–69, 1989.

[6] R. D. Brandt *et al.*, "Formulas for calculating supremal controllable and normal sublanguages," *Syst. Control Lett.*, vol. 15, no. 2, pp. 111–117, 1990.

[7] J. Komenda and J. van Schuppen, "Control of discrete-event systems with partial observations using coalgebra and coinduction," *Discrete Event Dynamic Syst.*, vol. 15, no. 3, pp. 257–315, 2005.

[8] S. Takai and T. Ushio, "Effective computation of an Lm(G)-closed, controllable, observable sublanguage arising in supervisory control," *Syst. Control Lett.*, vol. 49, no. 3, pp. 191–200, 2003.

[9] H. Cho and S. I. Marcus, "Supremal and maximal sublanguages arising in supervisor synthesis problems with partial observations," *Math. Syst. Theory*, vol. 22, no. 3, pp. 177–211, 1989.

[10] J. Fa, X. Yang, and Y. Zheng, "Formulas for a class of controllable and observable sublanguages larger than the supremal controllable and normal sublanguages," *Syst. Control Lett.*, vol. 20, no. 1, pp. 11–18, 1993.

[11] M. Heymann and F. Lin, "On-line control of partially observed discrete event systems," *Discrete Event Dynamic Syst.*, vol. 4, no. 3, pp. 221–236, 1994.

[12] N. B. Hadj-Alouane, S. Lafortune, and F. Lin, "Centralized and distributed algorithms for on-line synthesis of maximal control policies under partial observation," *Discrete Event Dynamic Syst.*, vol. 6, no. 4, pp. 379–427, 1996.

[13] T. Ushio, "On-line control of discrete event systems with a maximally controllable and observable sublanguage," *IEICE Trans. Fundamentals*, vol. E82-A, no. 9, pp. 1965–1970, 1999.

[14] K. Rudie and W. M. Wonham, "Think globally, act locally: Decentralized supervisory control," *IEEE Trans. Autom. Control*, vol. 37, no. 11, pp. 1692–1708, Nov. 1992.

[15] T. S. Yoo and S. Lafortune, "A general architecture for decentralized supervisory control of discrete-event systems," *Discrete Event Dynamic Syst.*, vol. 12, no. 3, pp. 335–377, 2002.

[16] Y. Li and W. M. Wonham, "Controllability and observability in the state-feedback control of discrete-event systems," in *Proc. 27th IEEE Conf. Decision Control*, Austin, TX, USA, 1988, pp. 203–208.

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

[18] F. Lin and W. M. Wonham, "Supervisory control of timed discrete-event systems under partial observation," *IEEE Trans. Autom. Control*, vol. 40, no. 3, pp. 558–562, Mar. 1995.

[19] S. Takai and T. Ushio, "A new class of supervisors for timed discrete event systems under partial observation," *Discrete Event Dynamic Syst.*, vol. 16, no. 2, pp. 257–278, 2006.

[20] H. Marchand, O. Boivineau, and S. Lafortune, "Optimal control of discrete event systems under partial observation," in *Proc. 40th IEEE Conf. Decision Control*, Orlando, FL, USA, 2001, pp. 2335–2340.

[21] F. Lin and W. M. Wonham, "Decentralized control and coordination of discrete-event systems with partial observation," *IEEE Trans. Autom. Control*, vol. 35, no. 12, pp. 1330–1337, Dec. 1990.

[22] R. Su, J. H. van Schuppen, and J. E. Rooda, "Aggregative synthesis of distributed supervisors based on automaton abstraction," *IEEE Trans. Autom. Control*, vol. 55, no. 7, pp. 1627–1640, Jul. 2010.

[23] J. Komenda, T. Masopust, and J. van Schuppen, "Synthesis of controllable and normal sublanguages for discrete-event systems using a coordinator," *Syst. Control Lett.*, vol. 55, no. 7, pp. 1627–1640, Jul. 2011.

[24] W. M. Wonham and P. J. Ramadge, "On the supremal controllable sublanguage of a given language," *SIAM J. Control Optimiz.*, vol. 25, no. 3, pp. 637–659, 1987.

[25] B. A. Davey and H. A. Priestley, *Introduction to Lattices and Order*. Cambridge, U.K.: Cambridge Univ. Press, 1990.

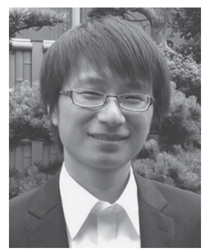
[26] L. Feng and W. M. Wonham, "Supervisory control architecture for discrete-event systems," *IEEE Trans. Autom. Control*, vol. 53, no. 6, pp. 1449–1461, 2008.

[27] K. Cai, R. Zhang, and W. M. Wonham, "Relative observability of discrete-event systems and its supremal sublanguages," 2013, Tech. Rep. [Online]. Available: http://arxiv.org/abs/1306.2422

[28] J. N. Tsitsiklis, "On the control of discrete-event dynamical systems," *Math. Control, Signals, Syst.*, vol. 2, no. 2, pp. 95–107, 1989.

[29] W. M. Wonham, "Design Software: XPTCT," Systems Control Group, Elect. Comput. Eng. Dept., Univ. of Toronto, updated Jul. 1, 2014. [Online]. Available: http://www.control.toronto.edu/DES

[30] L. E. Holloway and B. H. Krogh, "Synthesis of feedback logic control for a class of controlled petri nets," *IEEE Trans. Autom. Control*, vol. 35, no. 5, pp. 514–523, May 1990.



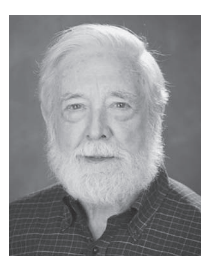
Kai Cai (M'12) received the B. Eng. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2006, the M.A.Sc. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 2008, and the Ph.D. degree in systems science from Tokyo Institute of Technology, Tokyo, Japan, in 2011.

He is currently an Associate Professor at Osaka City University, Osaka, Japan; preceding this position he was an Assistant Professor in the University of Tokyo from 2013 to 2014, and a Postdoctoral Fellow in the University of Toronto from 2011 to 2013. His research interests include distributed control of multi-agent systems, distributed control of discrete-event systems, and control architecture of complex networked systems.

Dr. Cai received the Best Paper Award at SICE in 2013, the Best Student Paper Award at the IEEE Multi-Conference on Systems and Control in 2010, and Young Author's Award of SICE in 2010.

Renyuan Zhang (M'12) received the B.Eng. degree and the Ph.D degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 2007 and 2013, respectively.

He is currently a Lecturer in the School of Automation, Northwestern Polytechnical University, Xi'an, China. From September 2011 to December 2012, he studied in the Department of Electrical and Computer Engineering, University of Toronto. His research interest is investigating delay-robust property in distributed control of untimed discrete-event systems and timed discrete-event systems.



W. M. Wonham (M'64–SM'76–F'77–LF'00) received the B.Eng. degree in engineering physics from McGill University, Montreal, QC, Canada, in 1956, and the Ph.D. degree in control engineering from the University of Cambridge, Cambridge, U.K., in 1961.

From 1961 to 1969, he was associated with several U.S. research groups in control. Since 1970, he has been a faculty member in systems control, with the Department of Electrical and Computer Engineering, University of Toronto. His research interests have included stochastic control and filtering, geometric multivariable control, and discrete-event systems. He is the author of *Linear Multivariable Control: A Geometric Approach* (Springer-Verlag, 3rd ed., 1985) and coauthor (with C. Ma) of *Nonblocking Supervisory Control of State Tree Structures* (Springer-Verlag, 2005).

Dr. Wonham is a Fellow of the Royal Society of Canada, a Life Fellow of the IEEE, and a Foreign Associate of the (U.S.) National Academy of Engineering. In 1987 he received the IEEE Control Systems Science and Engineering Award and in 1990 was Brouwer Medallist of the Netherlands Mathematical Society. In 1996 he was appointed University Professor in the University of Toronto, and in 2000 University Professor Emeritus.