Online Supervisory Control of Networked Discrete Event Systems With Control Delays

Zhaocong Liu, *Student Member, IEEE*, Xiang Yin, *Member, IEEE*, Shaolong Shu, *Senior Member, IEEE*, Feng Lin, *Fellow, IEEE*, and Shaoyuan Li, *Senior Member, IEEE*

***Abstract*—We investigate state estimation and safe controller synthesis for networked discrete-event systems (DESs), where supervisors send control decisions to plants via communication channels subject to communication delays. Previous works on state estimation of networked DES are based on the open-loop system without utilizing the knowledge of the control policy. In this article, we propose a new approach for online estimation and control of networked DES with control delays. We first propose a new state estimation algorithm for the closed-loop system utilizing the information of control decision history. The proposed state estimation algorithm can be implemented recursively upon the occurrence of each new observable event. Then we investigate how to predict the effect of control delays in order to calculate a control decision online at each instant. We show that the proposed online supervisor can be updated effectively and the resulting closed-loop behavior is safe. Furthermore, we compare the proposed online supervisor with the predictive supervisor proposed in the literature and show that our proposed online supervisor is more permissive than predictive supervisor in the sense of language inclusion.**

***Index Terms*—Control delay, discrete-event systems (DES), networked control systems, supervisory control.**

I. Introduction

A. Motivation

IN THIS article, we investigate the problem of supervisory control of discrete-event systems (DES). Supervisory control is a widely used approach for synthesizing controllers/supervisors with formal correctness guarantees of the desired specifications [38]. In many modern applications, supervisors are connected to the plants via communication networks. Such networked information structures provide more flexible ways for controlling DES, e.g., one is allowed to implement the supervisor on the cloud such that more powerful computational resources can be utilized to handle complex specifications. On the other hand, the networked information structure also brings many new research challenges, e.g., the effectiveness of communication delays/losses [7], [27], [35], [47], [48] and the security issue [9], [12], [39]. Therefore, networked DESs have drawn considerable attention in the past few years in the DES literature; see, e.g., [1], [18], [19], [22], [29], [37], [39], [44]–[46].

B. Related Works

The study of supervisory control of networked DES dates back to the work of Balemi [4], where the robustness of supervisors under communication delays was investigated. In [20] and [21], Park and Cho also studied the supervisory control problem of networked DES. In particular, it is assumed that each controllable event is disabled by default and it is enabled when sufficient information is obtained. In [24], the authors investigate the supervisory control problem in the setting where the supervisor and the plant cannot be synchronized perfectly due to communication delays and losses. In [45] and [46], Zgorzelski and Lunze used extended I/O automata to investigate the tracking control of networked DES and applied the solution to handling systems. Recently, Lin [16] proposed a general framework for supervisory control of networked DES. In particular, it considers communication delays and losses in both control channels and observation channels. Necessary and sufficient conditions, termed as networked controllability and networked observability, were provided for the existence of a nonpredictive supervisor that exactly achieves a given specification language.

Following the framework of Lin, many works on control of networked DES have been done in the literature in the past few years [13], [30], [31], [33], [50]. Particularly, in [33], the authors proposed a predictive supervisor and showed that the specification language is achievable if and only if the predictive supervisor can do so. The predictive supervisor has also been

Manuscript received May 5, 2020; revised October 31, 2020; accepted May 1, 2021. Date of publication May 14, 2021; date of current version April 26, 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0101700, and in part by the National Natural Science Foundation of China under Grants 62061136004, 61803259, 61833012, 61773287, and 62073242. Recommended by Associate Editor J. Komenda. *(Corresponding author: Xiang Yin.)*

Zhaocong Liu, Xiang Yin, and Shaoyuan Li are with the Department of Automation, Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zhaocongl@sjtu.edu.cn; yinxiang@sjtu.edu.cn; syli@sjtu.edu.cn).

Shaolong Shu and Feng Lin are with the School of Electronics and Information Engineering, Tongji University, Shanghai 201804, China, and also with the Department of Electrical and Computer Engineering, Wayne State University, Detroit MI 48202 USA (e-mail: flin@wayne.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAC.2021.3080495.

Digital Object Identifier 10.1109/TAC.2021.3080495

extended to modular systems by [13], where the monolithic system consists of a set of local modules that run concurrently. In [32], deterministic control of networked DES is studied in order to eliminate language uncertainty by enforcing the large language and the small language to be equal. The decentralized supervisory control of networked DES is investigated in [30], where the plant is controlled by a set of local supervisors that send local information to a fusion center in order to obtain a global decision. In [3], [23],[25], and [49], supervisory control of networked DES has also been extended to the setting of timed DESs. An approach that transforms the networked control problem to the nonnetworked setting under uncertain conditions was proposed in [50]. Networked supervisory control theory has also been applied to railway traffic control systems [16] and manufacturing systems [46].

In many applications, it is very difficult to achieve a given specification language exactly, i.e., *supervisor existence problem* is not solvable. Therefore, one is interested in *synthesizing* a supervisor such that the closed-loop language is a sublanguage of the desired specification. This problem is referred to as the *supervisor synthesis problem* in the literature, which is one of the central problems in the supervisory control theory [6], [10], [11], [14], [15], [34], [41]–[43]. The goal of the supervisor synthesis problem is to restrict the behavior of the system within the safety language but as permissive as possible. In the context of control of networked DES, in [2],[26], [28], [36], and [40], supervisor synthesis under observation delays and losses were investigated. For example, in [36] and [40], Mealy automata were used to model unreliable and nondeterministic observations. To our knowledge, however, the supervisor synthesis problem with control delays has not yet been fully investigated in the literature. For example, the work of [31] tackles the supervisor synthesis problem for safety specification under both control and observation delays. However, the approach in [31] requires to restrict the solution space *a priori*, while the solution space is infinite in general. When the specification language cannot be exactly achieved, the predictive supervisor proposed in [33] can also be used to solve the synthesis problem for safety. However, the control decision of the predictive supervisor is very conservative making the closed-loop behavior unnecessarily restrictive.

C. Our Results

In this article, we investigate the *supervisor synthesis problem* for networked DESs with safety specifications. Specifically, we focus on the case of control delays and assume that the system is only partially observed (see Fig. 1). It is worth noting that the synthesis problem under control delays is quite different from the case of observation delays. Particularly, in the standard supervisory control problem, the effective control decision can be updated immediately once a new event is observed. However, due to control delays, in the networked setting, the control decision issued currently may affect the closed-loop behavior in the future as the plant may still use a previous control decision issued by the supervisor. This issue has to be taken into account in the synthesis problem, which is also the main difficulty in the networked synthesis problem.

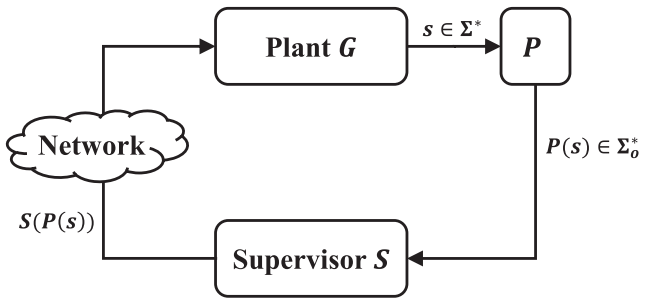

Fig. 1. Supervisory control of networked DES with control delays.

To solve this supervisor synthesis problem, we proposed a novel online control algorithm for networked DES with control delays. The proposed online control algorithm essentially consists of three parts: the state estimation part, the state prediction part, and the decision-making parting. The state estimation part aims to estimate the current state of the system based on both the observation and the control decision histories. To this end, a new concept called the extended state is proposed that augments the original plant state with the control channel configuration. We show that the extended state is the real "state" in the entire dynamic system as it also encodes the effect of previous control decision into the plant state. We propose an approach for estimating the extended state of the closed-loop system recursively online. Then, based on the estimated extended state, we discuss how to choose a control decision online by predicting its effect in the future. Specifically, we require that the selected control decision should not only be locally safe but also safe in the future before its effect expires. Finally, we integrate the state estimation, state prediction, and control selection techniques into an interactive online control algorithm and show its correctness. The general idea of online supervisory control was originally studied for standard DES without control delays; see, e.g., [5] and [11]. To the best of our knowledge, online supervisory control has never been applied to networked DES with communication delays.

Then we further compare the proposed online supervisor with the predictive supervisor proposed in [33]. It was shown in [33] that 1) the predictive supervisor can exactly achieve the specification language when the system is controllable and networked observable; and 2) it is still safe when the specification cannot be achieved. In fact, the predictive supervisor also contains a state estimation part and a state prediction part for the purpose of decision making. However, there is a significant difference between the estimation/prediction techniques in our work and those in [33]. Specifically, in the predictive supervisor, both the state estimation and the state prediction are based on the *open-loop system* without control. Such strategies are easy to implement, but the information of the closed-loop control policy is not used. Therefore, the state estimation and the state prediction used in the predictive supervisor coincide with our strategies only when the specification can be exactly achieved. In general, the estimated and predicted states of the predictive supervisor are overly conservative. On the other hand, both our state estimation and our state prediction techniques utilize the information of control decision history, which make the estimated/predicted

state more precise. Consequently, the online supervisor is more permissive than the predictive supervisor when the specification language cannot be exactly achieved.

D. Organization

The remainder of this article is organized as follows. In Section II, we introduce some necessary notations and formulate the problem to be solved. Online state estimation and state prediction techniques are provided in Sections III and IV, respectively. Also, we formally present the online control algorithm and prove its correctness in Section IV. In Section V, we further present a modified online control algorithm and show that it always outperforms the predictive supervisor proposed in the literature. Finally, we conclude the article by Section VI. Preliminary and partial versions of some of the results in this article are presented in [17]. Compared with [17], this article consists of proofs omitted in [17] as well as detailed explanations and example. Also, we provide a comparison between the proposed online supervisor with the predictive supervisor and show that the online supervisor can always outperform the predictive supervisor in terms of the permissiveness.

II. Preliminaries and Problem Formulation

Let Σ be a finite set of events. A string is a finite sequence of events. We denote by Σ^* the set of all strings over Σ including the empty string ϵ. A language $L \subseteq \Sigma^*$ is a set of strings. The prefix-closure of a language L is defined by $\overline{L} = \{w \in \Sigma^* : \exists t \text{ s.t. } wt \in L\}$. The concatenation of two languages $L_a, L_b \subseteq \Sigma^*$ is defined by $L_a L_b = \{s_a s_b \in \Sigma^* : s_a \in L_a, s_b \in L_b\}$.

For any string $s = \sigma_1 \sigma_2 \dots \sigma_n, \sigma_i \in \Sigma$, we denote by $|s|$ its length, i.e., $|s| = n$. Also, we denote by s_{-i} the string obtained by removing the last i events in s, i.e., $s_{-i} = \sigma_1 \dots \sigma_{|s|-i}$; we define $s_{-i} = \epsilon$, if $|s| \leq i$. For any natural number $N \in \mathbb{N}$, we denote by $[0, N]$ the set of natural numbers from 0 to N.

A DES is modeled by a finite-state automaton

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

where Q is a finite set of states, Σ is a finite set of events, $\delta : Q \times \Sigma \to Q$ is a partial transition function, and q_0 is the initial state. For any states $q, q' \in Q$ and event $\sigma \in \Sigma$, $\delta(q, \sigma) = q'$ implies that there exists a transition from q to q' labeled with event σ. The transition function is also extended to $\delta : Q \times \Sigma^* \to Q$ in the usual manner; see, e.g., [8]. For the sake of simplicity, we write $\delta(q, s)$ as $\delta(s)$ if $q = q_0$. Then the language generated by G is defined as $\mathcal{L}(G) = \{s \in \Sigma^* : \delta(s)!\}$, where "!" means "is defined."

In many situations, the original system G may not satisfy some desired specification. Therefore, the supervisory control theory was introduced in order to restrict the system's behavior such that the closed-loop system fulfills the specification. In the supervisory control framework, the event set Σ is partitioned as

$$\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc} = \Sigma_o \dot{\cup} \Sigma_{uo}$$

where Σ_c and Σ_{uc} denote the set of controllable events and the set of uncontrollable events, respectively. Similarly, Σ_o and Σ_{uo} denote the set of observable events and the set of unobservable events, respectively. Note that, in general, there is no relationship between controllable events and observable events.

We define $\Gamma = \{\gamma \in 2^\Sigma : \Sigma_{uc} \subseteq \gamma\}$ as the set of control decisions. That is, a supervisor cannot disable uncontrollable events. The natural projection $P : \Sigma^* \to \Sigma_o^*$ is defined recursively as follows: For any $s \in \Sigma^*, \sigma \in \Sigma$, we have

$$P(\epsilon) = \epsilon \text{ and } P(s\sigma) = \begin{cases} P(s)\sigma & \text{if } \sigma \in \Sigma_o \\ P(s) & \text{if } \sigma \in \Sigma_{uo}. \end{cases} \quad (1)$$

That is, upon the occurrence of a string $s \in \Sigma^*$, the supervisor can only be observed as $P(s) \in \Sigma_o^*$.

A supervisor is a mechanism that enables/disables events dynamically based on its observation. Formally, a supervisor is a mapping

$$S : P(\mathcal{L}(G)) \to \Gamma.$$

That is, the supervisor decides to enable events in $S(P(s))$ when string s is generated by the system.

In the networked setting, the supervisor needs to send its control decisions to the plant via control channels, where communication delays may occur. In this setting, the plant may still use a previous control decision even when a new control decision has been issued. We assume that the communication delays are bounded by a nonnegative integer N_c. In practice, the delay bound N_c can be identified by, e.g., communication experiments. Then the language generated by the closed-loop system (subject to control delays), denoted by $\mathcal{L}(S/G)$, is defined recursively as follows [16].

1) $\epsilon \in \mathcal{L}(S/G)$.
2) For any $s \in \Sigma^*$ and $\sigma \in \Sigma$, we have $s\sigma \in \mathcal{L}(S/G)$ if and only if
   1) $s\sigma \in \mathcal{L}(G)$;
   2) $s \in \mathcal{L}(S/G)$;
   3) $\sigma \in S(P(s)) \cup S(P(s_{-1})) \cup \dots \cup S(P(s_{-N_c}))$.

Remark 1: Intuitively, the definition of $\mathcal{L}(S/G)$ says that, at each instant, the plant may use any control decision issued by the supervisor in the previous N_c steps due to control delays. This language is also referred to as the "large language" (or the upper bound language) in the literature [32]. Note that this language is essentially an overapproximation of the actual behavior of the networked system and may contain some strings that are not feasible; see, e.g., [33] for detailed discussions. On the other hand, this definition provides a relatively simple language-based characteristic and is very suitable for the purpose of safety as it is conservative in the sense that it captures all possible actions the supervisor may take under control delays. Our article will also focus on the synthesis of safe supervisor; this is why we adopt this definition of the closed-loop language.

As we mentioned earlier, our goal is to design a supervisor such that the closed-loop system satisfies some requirement. Here, we consider a *safety specification*. Specifically, we assume that the specification is given as a legal language $K \subseteq \mathcal{L}(G)$ and we want to make sure that the behavior of the closed-loop system is within the legal language. Then we formulate the safety control problem with control delays (SCPCD) that we solve in this article. As we have discussed earlier, we consider the synthesis problem under the following assumptions.

1) The observation channel is perfect, i.e., each occurrence of observable event can be received by the supervisor instantly.
2) The control channel is subject to communication delays and the number of delays (counted by event occurrences) is upper bounded by a nonnegative integer N_c.
3) The initial control decisions is not subject to control delays, which can be, for example, embedded in actuators of the plant initially.

Problem 1: (SCPCD) Let G be a networked DES with control delays bounded by N_c. Let $K \subseteq \mathcal{L}(G)$ be a safety specification language. Find a safe supervisor S such that $\mathcal{L}(S/G) \subseteq K$.

Without loss of generality, we assume hereafter that K is recognized by a *strict subautomaton* of G, denoted by $H = (Q_H, \Sigma, \delta_H, q_0)$. That is, $Q_H \subseteq Q, \mathcal{L}(H) = K \subseteq \mathcal{L}(G)$, and for any string $s \in \mathcal{L}(G)$, we have $s \in K$ if and only if $\delta(q_0, s) \in Q_H$. We define

$$Q_{\text{good}} = \{q \in Q_H : \forall s \in \Sigma_{uc}^* \text{ s.t. } \delta(q, s) \in Q_H\}$$

as the set of states that cannot reach a state in $Q \setminus Q_H$ via uncontrollable transitions. We assume that $q_0 \in Q_{\text{good}}$. Therefore, to guarantee safety, we need to make sure that the system can only reach states in Q_{good}.

Remark 2: SCPCD can be solved by designing a supervisor that always disables all controllable events. However, this solution is not very interesting as the closed-loop behavior is very restrictive. Let S and S' be two safe supervisors. We say that S' is more permissive than S if $\mathcal{L}(S/G) \subset \mathcal{L}(S'/G)$. Therefore, we not only want to find a solution to SCPCD but also want the solution to be as permissive as possible.

Remark 3: In [33], the authors have investigated under what condition the specification language K can be *exactly* achieved subject to control delays. This problem is usually referred to as the *supervisor existence problem* in the literature [8]. Specifically, it has been shown that K can be exactly achieved if and only if the specification is controllable and networked observable. Our problem setting implicitly assumes that the existence conditions are not satisfied and we need to consider the *synthesis problem* rather than the existence problem.

III. Online State Estimation Under Control Delays

In the partial observation setting, in order to make control decision at each instant, the supervisor needs to *estimate* the set of all possible states the system can be in currently based on all information available. Formally, let G be a system, S be a supervisor (with control delays bounded by N_c), and $\alpha \in P(\mathcal{L}(S/G))$ be an observable string. Then the *state estimate* upon the occurrence of α is defined by

$$E_S(\alpha) = \{q \in Q : \exists s \in \mathcal{L}(S/G) \text{ s.t. } P(s) = \alpha \wedge \delta(s) = q\}.$$

Note that we use subscript "S" in $E_S(\cdot)$ in order to emphasize that we are considering the state estimate of the *closed-loop system* controlled by supervisor S. When supervisor S is given or we know *a priori* that the supervisor will exactly achieve a given language, the state estimate can be computed by constructing the (networked) observer based on the closed-loop behavior $\mathcal{L}(S/G)$ [16]. However, this state estimation technique essentially utilizes the entire functionality of the supervisor including the future control actions, which are unknown in the synthesis problem. In order to synthesize control decisions effectively, we need to estimate the state of the system *online* only based on the information available up to the current instant.

In order to state our online state estimation algorithm, first, we introduce the concept of *channel configuration*.

Definition 1: Let G be a networked DES with control delays bounded by N_c. A *channel configuration* is a set of pairs in the form of

$$\theta = \{(\gamma_1, n_1), (\gamma_2, n_2), \ldots, (\gamma_k, n_k)\}$$

where each $\gamma_i \in \Gamma$ is a control decision and each $n_i \in [0, N_c]$ is a nonnegative integer smaller than or equal to N_c. We denote by $\Gamma(\theta)$ the union of all control decision components in θ, i.e., $\Gamma(\theta) = \cup_{i=1,\ldots,k} \gamma_i$. Finally, we denote by $\Theta \subseteq 2^{\Gamma \times [0, N_c]}$ the set of all channel configurations.

Intuitively, each channel configuration specifies the control decisions that remained in the control channel and their timing information. That is, (γ_i, n_i) means that decision γ_i is delayed in the control channel and will still be effective for the next n_i steps. Essentially, the channel configuration models the "state" of the control channel. Therefore, in our setting, the true "state" of the closed-loop control system consists of both the state of the plant and the channel configuration, and we call this the *extended state*.

Definition 2: Let G be a networked DES with control delays bounded by N_c. An *extended state* is a plant state augmented with a channel configuration in the form of $\tilde{q} = (q, \theta)$, where $q \in Q$ and $\theta \in \Theta$. We define $\tilde{Q} := Q \times \Theta$ as the set of all extended states.

To precisely estimate the state of the system, we should not only track all possible states of the plant but also track the channel configuration of each possible state since delayed control decisions may affect the behavior of the system in the future. That is, we want to estimate all possible extended states based on the information available. Next, we describe how the extended state estimate evolves when new information is obtained (either when a new event is observed or when a new control decision is issued).

Let $\theta \in \Theta$ be a channel configuration. We define the "next" operator $\text{NX} : \Theta \to \Theta$ as follows: for any $\theta \in \Theta$,

$$\text{NX}(\theta) = \{(\gamma, n-1) \in \Gamma \times \mathbb{N} : (\gamma, n) \in \theta, n \geq 1\}.$$

That is, $\text{NX}(\theta)$ decreases the timing index of each control decision in θ by one unit and it only keeps control decisions whose timing indices are nonnegative (which means that the delay has not yet expired).

Now we are ready to present how the extended state estimate can be updated recursively online based on new information obtained. Suppose that our current estimation of the extended state of the system is $x \in 2^{\tilde{Q}}$. Then we need to update this set for the following two scenarios:

1) a new control decision $\gamma \in \Gamma$ is issued;
2) a new observable event $\sigma \in \Sigma_o$ is observed.

We formalize the estimation updating procedures for the above two scenarios by operators networked unobservable reach (NUR) : $2^{\tilde{Q}} \times \Gamma \to 2^{\tilde{Q}}$ and networked observable reach (NOR) : $2^{\tilde{Q}} \times \Sigma_o \to 2^{\tilde{Q}}$, respectively, as follows.

Definition 3: Let $x \in 2^{\tilde{Q}}$ be a set of extended states and $\gamma \in \Gamma$ be a control decision. Then the NUR of x under γ, denoted by $\text{NUR}_\gamma(x)$, is defined recursively as follows.

1) For any extended state $(q, \theta) \in x$, we have

$$(q, \theta \cup \{(\gamma, N_c)\}) \in \text{NUR}_\gamma(x). \tag{2}$$

2) For any extended state $(q, \theta) \in \text{NUR}_\gamma(x)$ and any unobservable event $\sigma \in \Sigma_{uo}$, we have

$$\begin{aligned} &\sigma \in \Gamma(\theta) \text{ and } \delta(q, \sigma)! \\ &\Rightarrow (\delta(q, \sigma), \text{NX}(\theta) \cup \{(\gamma, N_c)\}) \in \text{NUR}_\gamma(x). \end{aligned} \tag{3}$$

Intuitively, $\text{NUR}_\gamma(x)$ is the updated extended state estimate immediately after a new control decision γ (but before the occurrence of the next observable event) based on the latest state estimate x. The update is implemented by searching all extended states that can be reached unobservably from x. More specifically, first, we add the latest control decision γ and its timing index N_c to each extended state in x. Then, for each extended state $(q, \theta) \in \text{NUR}_\gamma(x)$, we need to consider all unobservable and feasible events from x. By feasible, we mean that the event is defined at q and is enabled by some control decision in the control channel, i.e., $\sigma \in \Gamma(\theta)$. For such (q, θ) and σ, we then add its successor extended state $(\delta(q, \sigma), \text{NX}(\theta) \cup \{(\gamma, N_c)\})$ to $\text{NUR}_\gamma(x)$. The second component of the successor state is $\text{NX}(\theta) \cup \{(\gamma, N_c)\}$ since 1) the execution of σ will decrease the timing index of each previous control decision in θ by one unit; and 2) we also need to add the current control decision with its timing index N_c.

Definition 4: Let $x \in 2^{\tilde{Q}}$ be a set of extended states and $\sigma \in \Sigma_o$ be an observable event. Then the NOR of x upon the occurrence of σ, denoted by $\text{NOR}_\sigma(x)$, is defined by

$$\text{NOR}_\sigma(x) = \{(\delta(q, \sigma), \text{NX}(\theta)) \in \tilde{Q} : (q, \theta) \in x, \sigma \in \Gamma(\theta)\}. \tag{4}$$

Intuitively, $\text{NOR}_\sigma(x)$ is the set of extended state estimate that can be reached immediately after observing σ but before the next control decision is issued, based on the latest state estimate x. Therefore, for each state $(q, \theta) \in x$ considered, event σ must be feasible in G and be enabled by some control decision in the control channel, i.e., $\sigma \in \Gamma(\theta)$. Upon the occurrence of σ, the plant state will be updated and we also need to decrease the timing index of each control decision in θ by one unit.

We use the following example to illustrate how NUR and NOR are computed.

Example 1: Let us consider system G shown in Fig. 2 with $\Sigma_o = \{a, b\}, \Sigma_c = \Sigma = \{a, b, u_1, u_2, u_3\}$, and $N_c = 1$. Suppose that the current extended state estimate is $x_1 = \{(1, \{(\Sigma, 1)\})\}$, which means that the system is at state 1 and there exists a control decision $\gamma_0 = \Sigma$ that will still be effective in one step. From this instant, if we observe event a, then we have

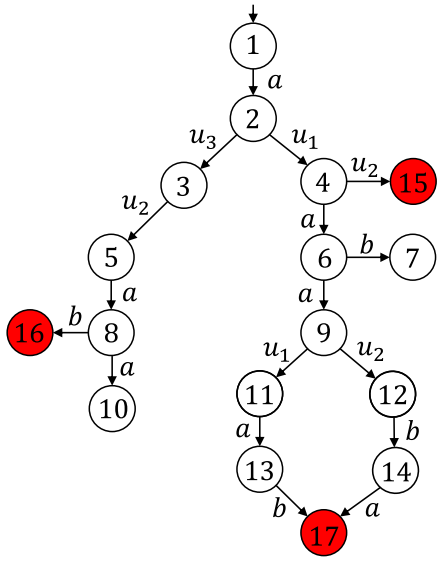

Fig. 2. System with $\Sigma_o = \{a, b\}, \Sigma_c = \Sigma$ and $N_c = 1$.

$$\hat{x}_2 = \text{NOR}_a(x_1) = \{(\delta(1, a), \text{NX}(\{(\gamma_0, 1)\}))\} = \{(2, \{(\gamma_0, 0)\})\}.$$

From $\hat{x}_2$, if we make control decision $\gamma_1 = \Sigma \setminus \{u_2\} = \{a, b, u_1, u_3\}$, then we have the following.

1) $(2, \{(\gamma_0, 0)\} \cup \{(\gamma_1, 1)\}) \in \text{NUR}_{\gamma_1}(\hat{x}_2)$.
2) Since $u_3 \in \Gamma(\{(\gamma_0, 0), (\gamma_1, 1)\})$ and $\delta(2, u_3) = 3$, we have $(3, \text{NX}(\{(\gamma_0, 0), (\gamma_1, 1)\}) \cup \{(\gamma_1, 1)\}) \in \text{NUR}_{\gamma_1}(\hat{x}_2)$.
3) Since $u_1 \in \Gamma(\{(\gamma_0, 0), (\gamma_1, 1)\})$ and $\delta(2, u_1) = 4$, we have $(4, \text{NX}(\{(\gamma_0, 0), (\gamma_1, 1)\}) \cup \{(\gamma_1, 1)\}) \in \text{NUR}_{\gamma_1}(\hat{x}_2)$.

Therefore, we have

$$\begin{aligned} \text{NUR}_{\gamma_1}(\hat{x}_2) = \{&(2, \{(\gamma_0, 0), (\gamma_1, 1)\}), (3, \{(\gamma_1, 0), (\gamma_1, 1)\}), \\ &(4, \{(\gamma_1, 0), (\gamma_1, 1)\})\}. \end{aligned}$$

We are now ready to present our online state estimation algorithm. The main idea is to employ $\text{NUR}_\gamma(\cdot)$ and $\text{NOR}_\sigma(\cdot)$ alternatively so that the extended state estimate can be updated recursively online. Formally, let G be a DES and S be a supervisor with control delays bounded by N_c. Let $\alpha \in P(\mathcal{L}(S/G))$ be an observable string generated by the closed-loop system. We define two sets $\hat{\mathcal{E}}_S(\alpha)$ and $\mathcal{E}_S(\alpha)$ recursively as follows: for any $\alpha \in \Sigma_o^*$ and $\sigma \in \Sigma_o$, we have

$$\hat{\mathcal{E}}_S(\epsilon) = \{(q_0, \emptyset)\} \tag{5}$$

$$\mathcal{E}_S(\alpha) = \text{NUR}_{S(\alpha)}(\hat{\mathcal{E}}_S(\alpha)) \tag{6}$$

$$\hat{\mathcal{E}}_S(\alpha\sigma) = \text{NOR}_\sigma(\mathcal{E}_S(\alpha)). \tag{7}$$

Intuitively, $\hat{\mathcal{E}}_S(\alpha)$ is the extended state estimate *immediately* after observing α and $\mathcal{E}_S(\alpha)$ is the extended state estimate after making control decision $S(\alpha)$ with unobservable reach included. Hereafter, we will formally show that $\mathcal{E}_S(\alpha)$ is indeed the extended state estimate of the closed-loop system upon the observation of α. A very important feature of $\mathcal{E}_S(\alpha)$ is that its

computation only utilizes the information available up to the instant of $\alpha = \sigma_1 \dots \sigma_n, \sigma_i \in \Sigma_o$, i.e., the following alternating sequence of control decisions and observations:

$$S(\epsilon)\sigma_1 S(\sigma_1)\sigma_2 \dots \sigma_n S(\sigma_1 \dots \sigma_n).$$

This makes online control synthesis possible as all information needed are available up to the current instant.

To state our result, for any $s \in \mathcal{L}(S/G)$, we define

$$\theta(s) = \{(\gamma, N_c - n) : 0 \le n \le \min\{N_c, |s|\}, \gamma = S(P(s_{-n}))\} \tag{8}$$

as the channel configuration upon the execution of s. Using this notation, for any $s \in \mathcal{L}(S/G)$, we have

$$\Gamma(\theta(s)) = S(P(s)) \cup S(P(s_{-1})) \cup \dots \cup S(P(s_{-N_c})). \tag{9}$$

The following result shows that our proposed state estimate $\mathcal{E}_S(\alpha)$ indeed correctly estimates the plant state together with its channel configuration.

Theorem 1: Let G be a DES and S be an arbitrary supervisor with control delays bounded by N_c. For any $\alpha \in P(\mathcal{L}(S/G))$, we have

$$\mathcal{E}_S(\alpha) = \{(\delta(s), \theta(s)) \in \tilde{Q} : \exists s \in \mathcal{L}(S/G) \text{ s.t. } P(s) = \alpha\}. \tag{10}$$

Proof: We prove by induction on the length of α.

Induction Basis: We consider the case of $|\alpha| = 0$, i.e., $\alpha = \epsilon$. Then we know that $\mathcal{E}_S(\epsilon) = \text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\})$. Next, we show that

$$\text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\}) = \{(\delta(s), \theta(s)) \in \tilde{Q} : s \in \mathcal{L}(S/G) \cap \Sigma_{uo}^*\}. \tag{11}$$

(LHS $\subseteq$ RHS) First, let $(q, \theta) \in \text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\})$ be an element of the LHS of (11). By Definition 3, we know that there exists a sequence of unobservable events $\sigma_1 \dots \sigma_n \in \Sigma_{uo}^*$ inducing the following extended states.

1) $(q_0, \theta_0) = (q_0, \emptyset \cup \{(S(\epsilon), N_c)\})$.
2) For any $k = 1, \dots, n$, we have
$(q_k, \theta_k) = (\delta(q_{k-1}, \sigma_k), \text{NX}(\theta_{k-1}) \cup \{(S(\epsilon), N_c)\})$
such that $\sigma_k \in \Gamma(\theta_{k-1})$ and $(q_n, \theta_n) = (q, \theta)$.

Since $\sigma_i \in S(\epsilon) \cap \Sigma_{uo}$ and $\delta(q_0, \sigma_1 \dots \sigma_n) = q$, we know that $\sigma_1 \dots \sigma_n \in \mathcal{L}(S/G) \cap \Sigma_{uo}^*$. Therefore, to show that (q, θ) is also an element of the RHS of (11), it remains to show that $\theta = \theta(s)$. Again, we show this by induction on the length of s. For $|s| = 0$, we know immediately that $\theta_0 = \{(S(\epsilon), N_c)\} = \theta(\epsilon)$. Now, we assume that $\theta_k = \theta(\sigma_1 \dots \sigma_k)$. Then we have

$$\begin{aligned}\theta_{k+1} &= \text{NX}(\theta_k) \cup \{(S(\epsilon), N_c)\} \\ &= \text{NX}(\theta(\sigma_1 \dots \sigma_k)) \cup \{(S(\epsilon), N_c)\} \\ &= \theta(\sigma_1 \dots \sigma_k \sigma_{k+1}).\end{aligned}$$

Therefore, we have LHS $\subseteq$ RHS.

(RHS $\subseteq$ LHS) Let (q, θ) be an element of the RHS of (11), i.e., there exists a sequence of unobservable events $s = \sigma_1 \dots \sigma_n \in \mathcal{L}(S/G) \cap \Sigma_{uo}^*$ such that $(\delta(s), \theta(s)) = (q, \theta)$. Hereafter, we show that $(q, \theta) \in \text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\})$. Still, we show this by induction on the length of s. When $|s| = 0$, we know that $(\delta(s), \theta(s)) = (q_0, \{(S(\epsilon), N_c)\}) \in \text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\})$. Now, we assume that $(\delta(s), \theta(s)) \in \text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\})$ for $|s| = k$. Then for the case of $|s| = k + 1$, i.e., $s = \sigma_1 \dots \sigma_{k+1}$, we know that

1) $\sigma_{k+1} \in \cup_{i=0,\dots,N_c} S(P((\sigma_1 \dots \sigma_k)_{-i})) = \Gamma(\theta(\sigma_1 \dots \sigma_k))$;
2) $\delta(\delta(\sigma_1 \dots \sigma_k), \sigma_{k+1})$!

Therefore, by Definition 3, we have

$$(\delta(s), \text{NX}(\theta(\sigma_1 \dots \sigma_k)) \cup \{(S(\epsilon), N_c)\}) \in \text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\}).$$

Also, since $\theta(s) = \text{NX}(\theta(\sigma_1 \dots \sigma_k)) \cup \{(S(\epsilon), N_c)\})$, we have $(\delta(s), \theta(s)) \in \text{NUR}_{S(\epsilon)}(\{(q_0, \emptyset)\})$ for $|s| = k + 1$.

Induction Step: Now, let us assume that (10) holds for $|\alpha| = m$ and we want to show that

$$\mathcal{E}_S(\alpha\sigma) = \{(\delta(s), \theta(s)) \in \tilde{Q} : \exists s \in \mathcal{L}(S/G) \text{ s.t. } P(s) = \alpha\sigma\} \tag{12}$$

where $\sigma \in \Sigma_o$ is a new observable event such that $\alpha\sigma \in P(\mathcal{L}(S/G))$. Similar to the induction basis, we still show this by two parts.

(LHS $\subseteq$ RHS) First, let $(q, \theta) \in \text{NUR}_{S(\alpha\sigma)}(\hat{\mathcal{E}}_S(\alpha\sigma))$ be an element of the LHS of (12). Note that $\mathcal{E}_S(\alpha\sigma) = \text{NUR}_{S(\alpha\sigma)}(\text{NOR}_\sigma(\mathcal{E}_S(\alpha)))$. By Definitions 3 and 4, we know that there exist an extended state $(q, \theta') \in \mathcal{E}_S(\alpha)$ and a sequence of events $\sigma\sigma_1 \dots \sigma_n$, where $\sigma_1 \dots \sigma_n \in \Sigma_{uo}^*$, inducing the following extended states.

1) $(q_1, \theta_1) = (\delta(q, \sigma), \text{NX}(\theta') \cup \{(S(\alpha\sigma), N_c)\})$.
2) For any $k = 1, \dots, n$, we have
$(q_{k+1}, \theta_{k+1}) = (\delta(q_k, \sigma_k), \text{NX}(\theta_k) \cup \{(S(\alpha\sigma), N_c)\})$
such that
a) $\sigma \in \Gamma(\theta'), \sigma_k \in \Gamma(\theta_k)$;
b) $(q_{n+1}, \theta_{n+1}) = (q, \theta)$.

Let $t \in \mathcal{L}(S/G)$ be a string that $P(t) = \alpha$ and $(\delta(t), \theta(t)) = (q, \theta')$. Next, we show the following by induction:

1) $t\sigma\sigma_1 \dots \sigma_k \in \mathcal{L}(S/G)$;
2) $\theta_{k+1} = \theta(t\sigma\sigma_1 \dots \sigma_k)$.

For $k = 0$, we have $\theta_k = \theta' = \theta(t)$. Since $q' = \delta(t)$, $\delta(q, \sigma) = q_1$, and $\Gamma(\theta') = \Gamma(\theta(t)) = \cup_{i=0,\dots,N_c} S(P(t_{-i}))$, we have $t\sigma \in \mathcal{L}(S/G)$. Also, we have

$$\begin{aligned}\theta_1 &= \text{NX}(\theta') \cup \{(S(\alpha\sigma), N_c)\} = \text{NX}(\theta(t)) \cup \{(S(\alpha\sigma), N_c)\} \\ &= \theta(t\sigma).\end{aligned} \tag{13}$$

Now we assume that the above two conditions hold for k and we consider the case of $k + 1$. Still, since $q_{k+1} = \delta(t\sigma\sigma_1 \dots \sigma_k)$, $\delta(q_{k+1}, \sigma_{k+1})$!, and $\sigma_{k+1} \in \Gamma(\theta_{k+1}) = \Gamma(\theta(t\sigma\sigma_1 \dots \sigma_k)) = \cup_{i=0,\dots,N_c} S(P((t\sigma\sigma_1 \dots \sigma_k)_{-i}))$, we have $s\sigma\sigma_1 \dots \sigma_{k+1} \in \mathcal{L}(S/G)$. Also, we have

$$\begin{aligned}\theta_{k+2} &= \text{NX}(\theta_{k+1}) \cup \{(S(\alpha\sigma), N_c)\} \\ &= \text{NX}(\theta(t\sigma\sigma_1 \dots \sigma_k)) \cup \{(S(\alpha\sigma), N_c)\} \\ &= \theta(t\sigma\sigma_1 \dots \sigma_k\sigma_{k+1}).\end{aligned} \tag{14}$$

Therefore, by taking $t\sigma\sigma_1 \dots \sigma_n$ as string s in the RHS of (12), we have that $(q, \theta) = (q_n, \theta_n)$ is also an element in the RHS of (12).

(RHS $\subseteq$ LHS) Let (q, θ) be an element of the RHS of (12), i.e., there exists a string $s \in \mathcal{L}(S/G)$ such that $P(s) = \alpha\sigma$ and $(\delta(s), \theta(s)) = (q, \theta)$. We write string s in the form of $s = t\sigma\sigma_1 \dots \sigma_n$, where $\sigma_1 \dots \sigma_n \in \Sigma_{uo}^*$ is the unobservable tail.

We define the following sequence of extended states recursively by the following.

1) $(q_1,\theta_1)=(\delta(t\sigma),\text{NX}(\theta(t))\cup\{(S(\alpha\sigma),N_c)\})$.
2) For any $k=1,\ldots,n$, we have
$(q_{k+1},\theta_{k+1})=(\delta(q_k,\sigma_k),\text{NX}(\theta_k)\cup\{(S(\alpha\sigma),N_c)\})$.

First, we note that $\theta_{i+1}=\theta(t\sigma\sigma_1\ldots\sigma_i)\forall i=0,\ldots,n$. Next, we show by induction that $(q_i,\theta_i)\in\mathcal{E}_S(\alpha\sigma)\forall i=1,\ldots,n+1$.

Initially, since $P(t)=\alpha$, by the induction hypothesis (on $|\alpha|$), we know that $(\delta(t),\theta(t))\in\mathcal{E}_S(\alpha)$. Since $t\sigma\in\mathcal{L}(S/G)$, we know that $\sigma\in\Gamma(\theta(t))$. Therefore, by Definition 4, we know that

$$(\delta(t\sigma),\text{NX}(\theta(t)))\in\text{NOR}_\sigma(\mathcal{E}_S(\alpha)).$$

Then by Definition 3, we know that

$$\begin{aligned}(q_1,\theta_1)&=(\delta(t\sigma),\text{NX}(\theta(t))\cup\{(S(\alpha\sigma),N_c)\})\\&\in\text{NUR}_{S(\alpha\sigma)}(\text{NOR}_\sigma(\mathcal{E}_S(\alpha)))=\mathcal{E}_S(\alpha\sigma).\end{aligned}\tag{15}$$

Now we assume that $(q_k,\theta_k)\in\mathcal{E}_S(\alpha\sigma)$ and we consider the case of $k+1$. Since $t\sigma\sigma_1\ldots\sigma_{k-1}\sigma_k\in\mathcal{L}(S/G)$, we have

$$\begin{aligned}\sigma_k&\in\cup_{i=0,\ldots,N_c}S(P((t\sigma\sigma_1\ldots\sigma_{k-1})_{-i}))\\&=\Gamma(\theta(t\sigma\sigma_1\ldots\sigma_{k-1}))=\Gamma(\theta_k).\end{aligned}\tag{16}$$

Again, since $(q_{k+1},\theta_{k+1})=(\delta(q_k,\sigma_k),\text{NX}(\theta_k)\cup\{(S(\alpha\sigma),N_c)\}$, by Definition 3, we have $(q_{k+1},\theta_{k+1})\in\mathcal{E}_S(\alpha\sigma)$.

Overall, we prove the equality in (10). ■

For any set of extended states $x=\{(q_1,\theta_1),\ldots,(q_n,\theta_n)\}$, we denote by $Q(x)$ the set of all states in its first component, i.e., $Q(x)=\{q_1,q_2,\ldots,q_n\}$. Then we have the following corollary of Theorem 1 by restricting our attention to the first component of x.

Corollary 1: Let G be a DES and S be an arbitrary supervisor with control delays bounded by N_c. For any $\alpha\in P(\mathcal{L}(S/G))$, we have $Q(\mathcal{E}_S(\alpha))=E_S(\alpha)$.

We illustrate our online state estimation procedure by the following example.

Example 2: Let us still consider the system G shown in Fig. 2 with $\Sigma_c=\Sigma,\Sigma_o=\{a,b\}$ and $N_c=1$. Suppose that the system is controlled by supervisor S defined by the following:

1) $S(\epsilon)=\gamma_0:=\Sigma$;
2) $S(a)=\gamma_1:=\Sigma\setminus\{u_2\}$.

We now compute the proposed extended state estimation $\mathcal{E}_S(a)$ and $E_S(a)$, respectively.

Initially, we have $\hat{\mathcal{E}}_S(\epsilon)=\{(1,\emptyset)\}$. After making the first control decision $S(\epsilon)=\gamma_0$, we have $\mathcal{E}_S(\epsilon)=\text{NUR}_{\gamma_0}(\hat{\mathcal{E}}_S(\epsilon))=\{(1,\{(\gamma_0,1)\})\}$. Then upon observing a, we have $\hat{\mathcal{E}}_S(a)=\text{NOR}_a(\mathcal{E}_S(\epsilon))=\{(2,\{(\gamma_0,0)\})\}$. After making the second control decision $S(a)=\gamma_1$, we further have $\mathcal{E}_S(a)=\text{NUR}_{\gamma_1}(\hat{\mathcal{E}}_S(a))=\{(2,\{(\gamma_0,0),(\gamma_1,1)\}),(3,\{(\gamma_1,0),(\gamma_1,1)\}),(4,\{(\gamma_1,0),(\gamma_1,1)\})\}$. Note that state 5 is not contained in $\mathcal{E}_S(a)$ since $u_2\notin\Gamma(\{(\gamma_1,0),(\gamma_1,1)\})=\{a,b,u_1,u_3\}$.

On the other hand, by definition of $E_S(\cdot)$ and $\mathcal{L}(S/G)$, we have $\{\epsilon,a,au_1,au_3\}\subseteq\mathcal{L}(S/G)$. However, $au_3u_2\notin\mathcal{L}(S/G)$ since $u_2\notin S(P(au_3))\cup S(P((au_3)_{-1}))=\gamma_1$. Therefore, we have $E_S(a)=\{\delta(s)\in Q:s\in\mathcal{L}(S/G)\text{ s.t. }P(s)=a\}=\{2,3,4\}$. This is consistent with our result that $E_S(a)=Q(\mathcal{E}_S(a))$.

IV. Online Control Algorithm

In the previous section, we have shown how to compute the state estimate of the system online in the presence of control delays. In this section, we will discuss how to choose a safe control decision at each instant based on the state estimate.

A. Uncontrollable State Prediction

Note that in the standard supervisory control framework without control delays, the choice of control decision at each instant will only affect reachable states until the occurrence of the next observable event since the control decision can be updated immediately. However, in our setting, the current control decision may affect reachable states in the next N_c steps due to control delays. Therefore, to precisely evaluate the effect of a control, we need to *predict* all possible states that can be reached in the presence of control delays. To this end, we introduce the concept of *uncontrollable state prediction* (USP).

Let $\theta=\{(\gamma_1,n_1),(\gamma_2,n_2),\ldots,(\gamma_k,n_k)\}$ be a channel configuration and $m\in[0,N_c]$ be a nonnegative integer. We denote by $\Gamma_{\geq m}(\theta)$ the union of all control decisions that will still be effective after m steps, i.e.,

$$\Gamma_{\geq m}(\theta)=\bigcup_{i\in\{1,\ldots,k\}:n_i\geq m}\gamma_i.\tag{17}$$

Clearly, we have $\Gamma_{\geq 0}(\theta)=\Gamma(\theta)$ and $\Gamma_{\geq m}(\theta)\subseteq\Gamma(\theta)$. Then we define the *uncontrollable language* from θ by

$$L_{uc}(\theta):=\overline{\Gamma_{\geq 0}(\theta)\Gamma_{\geq 1}(\theta)\cdots\Gamma_{\geq N_c}(\theta)}$$

where the second part is the prefix-closure of the concatenation of event sets $\Gamma_{\geq i}(\theta)$ from $i=0$ to $i=N_c$. Intuitively, if the current channel configuration is θ, then we cannot prevent any string in $L_{uc}(\theta)$ from happening due to existing delayed control decisions in the channel.

Definition 5: Let $\tilde{q}=(q,\theta)\in\tilde{Q}$ be an extended state, then the *USP* of $\tilde{q}$, denoted by $\text{USP}(\tilde{q})$, is defined by

$$\text{USP}(\tilde{q})=\{\delta(q,s)\in Q\colon s\in L_{uc}(\theta)\}.\tag{18}$$

The USP is also extended to a set of extended states $x\in 2^{\tilde{Q}}$ by $\text{USP}(x)=\cup_{\tilde{q}\in x}\text{USP}(\tilde{q})$.

The intuition of the USP is similar to that of the uncontrollable language. It essentially captures the set of states we cannot prevent from reaching from x no matter what control decisions we take in the future. We illustrate this concept by the following example.

Example 3: Let us still consider the system G shown in Fig. 2 with $\Sigma_c=\Sigma,\Sigma_o=\{a,b\}$ and $N_c=1$. Note that there are three illegal states that should be avoided which are in the shape of solid red circles, i.e., $Q\backslash Q_H=\{15,16,17\}$. Suppose we are now at extended state $\tilde{q}=(11,\{(\gamma,0),(\gamma,1)\})$, where $\gamma=\Sigma_o=\{a,b\}$, and we want to compute its uncontrollable state prediction $\text{USP}(\tilde{q})$. First, we have

$$L_{uc}(\theta)=\overline{\Gamma_{\geq 0}(\theta)\Gamma_{\geq 1}(\theta)}=\overline{\{a,b\}\{a,b\}}$$

$$= \overline{\{aa, ab, ba, bb\}}.$$

Therefore, we have

$$\text{USP}(\tilde{q}) = \{\delta(11, s) \in Q \colon s \in L_{uc}(\theta)\} = \{11, 13, 17\}.$$

Therefore, we know that we will unavoidably reach illegal state 17 starting from such extended state $\tilde{q} = (11, \theta)$.

B. Choice of Control Decision

Suppose that string $\alpha \in \Sigma_o^*$ is observed and the extended state estimate of the system (before the latest unobservable reach) is computed as $\hat{\mathcal{E}}_S(\alpha) \in 2^{\tilde{Q}}$. Now, let us discuss how to choose a control decision at each instant for the purpose of safety. As we discussed above, once we choose control decision $\gamma \in \Gamma$ at $\hat{\mathcal{E}}_S(\alpha)$, the extended state estimate will be updated as $\mathcal{E}_S(\alpha) = \text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))$. Moreover, the system will possibly reach any state in $\text{USP}(\mathcal{E}_S(\alpha))$ no matter what control decisions we take in the future. Therefore, we say that a control decision $\gamma \in \Gamma$ is *safe* at $\hat{\mathcal{E}}_S(\alpha)$ if

$$\text{USP}(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))) \subseteq Q_{\text{good}}. \tag{19}$$

Therefore, to guarantee safety, we need to make sure that the control decision at each instant is safe; otherwise, the system may unavoidably reach an illegal state in $Q \setminus Q_H$.

So far, we have discussed how to estimate the state of the system recursively online and how to predict the effect of a control decision based on the current state estimate. Now, we combine the estimation and prediction techniques together to finally present our online control algorithm.

The online control procedure is formally provided in Algorithm ONLINE-CONTROL, which computes a control decision at each instant upon the occurrence of a new observable event. More specifically, the procedure starts from the initial state estimate $\hat{\mathcal{E}}_S(\epsilon) = \{(q_0, \emptyset)\}$. Then it wants to find a safe control decision γ at $\hat{\mathcal{E}}_S(\epsilon)$ in the sense of (19). Moreover, to achieve permissiveness, we want this control decision to be *maximal*, i.e., there does not exist another safe control decision γ' at $\hat{\mathcal{E}}_S(\epsilon)$ such that $\gamma \subset \gamma'$. Once the control decision is chosen, we update our state estimate to $\mathcal{E}_S(\epsilon)$ using the $\text{NUR}_\gamma(\cdot)$ operator and then wait for the occurrence of the next observable event. Once a new observable event σ is observed, first, we update the state estimate to $\hat{\mathcal{E}}_S(\sigma)$ using the $\text{NOR}_\sigma(\cdot)$ operator. Then, we go to line 7 to choose a safe control decision and repeat the above procedure indefinitely.

Remark 4: The maximal safety control decision in lines 2 and 7 of Algorithm 1 can be found by an "add-and-test" manner. Specifically, we can start from the set of uncontrollable events and then add a controllable event to it and test whether or not the resulting set is still safe. If so, we keep this event and repeat this procedure until no event can be added anymore. Similar procedure has been described in more detail in the literature; see, e.g., [5].

Let us use the following example to illustrate Algorithm 1.

Example 4: Let us return to the system G shown in Fig. 2 with $\Sigma_c = \Sigma, \Sigma_o = \{a, b\}$ and $N_c = 1$. The specification language K is generated by the subautomaton obtained by removing illegal states (shown in red solid circles) from G. For this example, we have $Q_H = Q_{good}$ as all events are controllable.

Algorithm 1: ONLINE-CONTROL (G, N_c, Q_{good}).

1 $\alpha \leftarrow \epsilon$, $\hat{\mathcal{E}}_S(\alpha) \leftarrow \{(q_0, \emptyset)\}$;
2 Find a maximal control decision $\gamma \in \Gamma$ such that

$$\text{USP}(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))) \subseteq Q_{good}$$

3 Make control decision $S(\alpha) \leftarrow \gamma$;
4 $\mathcal{E}_S(\alpha) \leftarrow \text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))$;
while *a new event* $\sigma \in \Sigma_o$ *is observed* **do**
5 $\quad \hat{\mathcal{E}}_S(\alpha\sigma) \leftarrow \text{NOR}_\sigma(\mathcal{E}_S(\alpha))$;
6 $\quad \alpha \leftarrow \alpha\sigma$;
7 $\quad$ Find a maximal control decision $\gamma \in \Gamma$ such that

$$\text{USP}(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))) \subseteq Q_{good}$$

8 $\quad$ Make control decision $S(\alpha) \leftarrow \gamma$;
9 $\quad \mathcal{E}_S(\alpha) \leftarrow \text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))$;

Next, we apply the proposed online control algorithm to compute control decision for the first three instants, i.e., $S(\epsilon)$, $S(a)$, and $S(aa)$.

Initially, the algorithm starts from $\hat{\mathcal{E}}(\epsilon) = \{(q_0, \emptyset)\}$ and we want to choose a maximal control decision $S(\epsilon) = \gamma_0$ such that $\text{USP}(\text{NUR}_{\gamma_0}(\hat{\mathcal{E}}_S(\epsilon))) \subseteq Q_{good}$. One can check that $\gamma_0 = \Sigma$ is such a maximal control decision. Then the state estimate is updated to $\mathcal{E}_S(\epsilon) = \text{NUR}_{\gamma_0}(\hat{\mathcal{E}}_S(\epsilon)) = \{(1, \{(\gamma_0, 1)\})\}$. If event a is observed, then we first update the state estimate to $\hat{\mathcal{E}}_S(a) = \text{NOR}_a(\mathcal{E}_S(\epsilon)) = \{(2, \{(\gamma_0, 0)\})\}$. Then we again want to find a maximal control decision $S(a) = \gamma_1$ such that $\text{USP}(\text{NUR}_{\gamma_1}(\{(2, \{(\gamma_0, 0)\})\})) \subseteq Q_{good}$. For this, we can choose $\gamma_1 = \{a, b, u_1, u_3\}$. Note that we cannot add event u_2 to γ_1 since illegal state 15 will be reached in the unobservable reach. Then the immediate state estimate upon the occurrence of the next observable event a is $\hat{\mathcal{E}}(aa) = \{(6, \{(\gamma_1, 0)\})\}$. By applying the online algorithm recursively again, we can obtain control decision $S(aa) = \gamma_2 = \Sigma$.

C. Correctness of Online Control Algorithm

In this subsection, we formally prove the correctness of the proposed online control algorithm. Specifically, we show that Algorithm 1 is what follows:

1) *recursively feasible*, i.e., we can always find a control decision satisfying the condition in lines 2 and 7 in the while-loop at each instant, no matter what events and control decisions are executed in the past;
2) *safe*, i.e., the resulting online supervisor will never reach an illegal state.

First, we show the recursive feasibility of Algorithm 1 by the following theorem.

Theorem 2: Algorithm 1 is recursively feasible. That is, at each instant, we can always find a control decision $\gamma \in \Gamma$ such that $\text{USP}(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))) \subseteq Q_{good}$.

Proof: To show this, it suffices to show that $\text{USP}(\text{NUR}_{\Sigma_{uc}}(\hat{\mathcal{E}}_S(\alpha))) \subseteq Q_{good}$. That is, at least disabling all controllable events is a valid choice at lines 2 and 7. We proceed by induction on the length of the observation string.

Induction Basis: Initially, we have $\alpha = \epsilon$ and $\hat{\mathcal{E}}_S(\epsilon) = \{(q_0, \emptyset)\}$. Therefore, for any extended state $(q, \theta) \in \text{NUR}_{\Sigma_{uc}}(\hat{\mathcal{E}}_S(\epsilon))$, we know that q is reached via some string in Σ_{uc}^* from q_0 and $\forall 0 \leq i \leq N_c : \Gamma_{\geq i}(\theta) = \Sigma_{uc}$, which implies that $L_{uc}(\theta) \subseteq \Sigma_{uc}^*$. Since $q_0 \in Q_{good}$, we have

$$\text{USP}((q, \theta)) \subseteq \{\delta(q_0, s) \in Q : s \in \Sigma_{uc}^*\} \subseteq Q_{good}.$$

Since (q, θ) is arbitrary, we have $\text{USP}(\text{NUR}_{\Sigma_{uc}}(\hat{\mathcal{E}}_S(\epsilon))) \subseteq Q_{good}$, which proves the induction basis.

Induction Step: Now, let $\alpha\sigma \in \Sigma_o^*$ be an observable sequence that can happen under the control of online supervisor, where $\sigma \in \Sigma_o$ and $|\alpha| = k$. We assume that the condition in (19) can always be fulfilled at each instant along α and we need to show that control decision Σ_{uc} can still fulfill this condition for the instant of $\alpha\sigma$. To this end, we assume, for the sake of contradiction, that

$$\text{USP}(\text{NUR}_{\Sigma_{uc}}(\text{NOR}_{\sigma}(\mathcal{E}_S(\alpha))) \not\subseteq Q_{\text{good}}. \tag{20}$$

This implies that there exists an extended state

$$(q, \theta) \in \text{NUR}_{\Sigma_{uc}}(\text{NOR}_{\sigma}(\mathcal{E}_S(\alpha))$$

such that $\text{USP}((q, \theta)) \not\subseteq Q_{\text{good}}$. That is, there exists a string

$$s \in L_{uc}(\theta) = \overline{\Gamma_{\geq 0}(\theta)\Gamma_{\geq 1}(\theta) \cdots \Gamma_{\geq N_c}(\theta)}$$

such that $\delta(q, s) \notin Q_{\text{good}}$.

Since $(q, \theta) \in \text{NUR}_{\Sigma_{uc}}(\text{NOR}_{\sigma}(\mathcal{E}_S(\alpha)))$, by the definitions of NUR and NOR, there exists a state $(\hat{q}, \hat{\theta}) \in \mathcal{E}_S(\alpha)$ and a string $\sigma_1\sigma_2 \ldots \sigma_m \in \Sigma_{uo}^*$ that induce the following sequence of extended states:

$$(\hat{q}, \hat{\theta}) \xrightarrow{\sigma} (q_1, \theta_1) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_m} (q_{m+1}, \theta_{m+1})$$

where

1) $\sigma \in \Gamma(\hat{\theta})$ and $(q_1, \theta_1) = (\delta(\hat{q}, \sigma), \text{NX}(\hat{\theta}) \cup \{(\Sigma_{uc}, N_c)\})$;
2) for any $1 \leq i \leq m$, we have $\sigma_i \in \Gamma(\theta_i)$ and $(q_{i+1}, \theta_{i+1}) = (\delta(q_i, \sigma_i), \text{NX}(\theta_i) \cup \{(\Sigma_{uc}, N_c)\})$;
3) $(q_{m+1}, \theta_{m+1}) = (q, \theta)$.

Then, for the above sequence, we have

$$\Gamma(\theta_i) = \begin{cases} \Gamma_{\geq i}(\hat{\theta}) & \text{if } i \leq N_c \\ \Sigma_{uc} & \text{if } i > N_c \end{cases}. \tag{21}$$

This implies that, starting from extended state $(q_{m+1}, \theta_{m+1}) = (q, \theta)$, we also have

$$\Gamma_{\geq i}(\theta_{m+1}) = \begin{cases} \Gamma_{\geq i}(\hat{\theta}) & \text{if } m+1+i \leq N_c \\ \Sigma_{uc} & \text{if } m+1+i > N_c. \end{cases} \tag{22}$$

Recall that $\delta(q_{m+1}, s) \notin Q_{\text{good}}$ and $s \in L_{uc}(\theta_{m+1})$. Therefore, we have

$$\sigma\sigma_1 \ldots \sigma_m s \in \overline{\Gamma_{\geq 0}(\hat{\theta}) \cdots \Gamma_{\geq N_c}(\hat{\theta})}\Sigma_{uc}^* = L_{uc}(\hat{\theta})\Sigma_{uc}^*.$$

Let $s' \in L_{uc}(\hat{\theta})$ be the longest prefix of $\sigma\sigma_1 \ldots \sigma_m s$ such that the last event is controllable. By the definition of Q_{good}, we know that $\delta(\hat{q}, s') \notin Q_{\text{good}}$. However, by the induction hypothesis, we have $\text{USP}(\mathcal{E}_S(\alpha)) \subseteq Q_{\text{good}}$, which implies that $\{\delta(\hat{q}, t) \in Q : t \in L_{uc}(\hat{\theta})\} \subseteq Q_{\text{good}}$. This is a contradiction. Therefore, we know that $\text{USP}(\text{NUR}_{\Sigma_{uc}}(\text{NOR}_{\sigma}(\mathcal{E}_S(\alpha))) \subseteq Q_{\text{good}}$, which completes the induction step. ■

Next, we show that the proposed online supervisor is safe. This result follows directly from the choice of control decision at each instant, which guarantees that no state in $Q \setminus Q_{\text{good}}$ can be reached.

Theorem 3: Let G be a DES and we denote by S_{online} the online supervisor defined in Algorithm 1. Then we have $\mathcal{L}(S_{\text{online}}/G) \subseteq K$.

Proof: We assume, for the sake of contradiction, that $\mathcal{L}(S_{\text{online}}/G) \not\subseteq K$. Let $s \in \mathcal{L}(S_{\text{online}}/G) \setminus K$ be a string violating the specification. Let $P(s) = \alpha$. We assume without loss of generality that $\alpha \neq \epsilon$; the case of $\alpha = \epsilon$ can be shown analogously. Since $\delta(q_0, s) \in E_S(\alpha)$ and $\delta(q_0, s) \notin Q_{\text{good}}$, there exists a longest prefix of α, denoted by α', such that $E_S(\alpha') \subseteq Q_{\text{good}}$. Note that $E_S(\epsilon) \subseteq Q_{\text{good}}$ is guaranteed by the initial choice of $S(\epsilon)$ when line 2 was executed for the first time. We denote by σ the following event of α' in α, i.e., $E_S(\alpha'\sigma) \not\subseteq Q_{\text{good}}$. By Corollary 1, we know that $Q(\mathcal{E}_S(\alpha'\sigma)) = E_S(\alpha'\sigma)$. Furthermore, by the definition of USP, we know that $Q(\mathcal{E}_S(\alpha'\sigma)) \subseteq \text{USP}(\mathcal{E}_S(\alpha'\sigma))$. By the choice of control decision $S(\alpha'\sigma)$ at line 7, we then have

$$E_S(\alpha'\sigma) = Q(\mathcal{E}_S(\alpha'\sigma)) \subseteq \text{USP}(\mathcal{E}_S(\alpha'\sigma)) \subseteq G_{\text{good}}.$$

However, this is a contradiction, which means that $\mathcal{L}(S_{\text{online}}/G) \subseteq K$. ■

V. Comparison With Predictive Supervisor

In the previous section, we have provided an online approach for solving SCPCD. In [33], a supervisor called *predictive supervisor* was proposed in order to solve the supervisor existence problem that aims to achieve K exactly. Moreover, the author also shows that when K is not exactly achievable, the synthesized supervisor is still safe, i.e., the predictive supervisor can also provide a solution to SCPCD.

In this section, we formally compare our solution with the predictive supervisor in the literature when it is used for the synthesis problem. Specifically, we show the following.

1) In general, the proposed online supervisor in Algorithm 1 is incomparable with the predictive supervisor.
2) With an additional criterion on the choice of control decision at each instant, which can always be fulfilled, the proposed online supervisor can outperform the predictive supervisor in terms of the permissiveness.

A. Review of Predictive Supervisor

First, we review the predictive supervisor proposed in [33] for the case of control delays.

Recall that the specification language K is recognized by a strict subautomaton $H = (Q_H, \Sigma, \delta_H, q_0)$, with $Q_{\text{good}} \subseteq Q_H$. We further assume, without loss of generality, that K is controllable, which implies $Q_{\text{good}} = Q_H$; otherwise, we can compute the supremal controllable sublanguage of K. Then we define the

observer of H by

$$H_{\text{obs}} = (Y, \Sigma_o, \xi, y_0)$$

where $Y \subseteq 2^{Q_H}$ is the set of states, $y_0 = \{\delta_H(q_0, w) \in Q_H : w \in \Sigma_{uo}^*\}$ is the initial state. The transition function $\xi : Y \times \Sigma_o \to Y$ is defined by the following: for any $y \in Y$ and $\sigma \in \Sigma_o$, we have

$$\xi(y, \sigma) = \text{UR}(\{\delta_H(q, \sigma) \in Q_H : q \in y\})$$

where $\text{UR}(\cdot)$ is the standard unobservable reach operator defined by the following: for any $y \in 2^{Q_H}$, we have

$$\text{UR}(y) := \{\delta_H(q, w) \in Q_H : q \in y, w \in \Sigma_{uo}^*\}.$$

For the sake of simplicity, we only consider the reachable part of H_{obs}. Essentially, the observer computes the current state-estimate based on H, i.e., for any string $\alpha \in P(\mathcal{L}(H))$, we have $\xi(y_0, \alpha) = E(\alpha)$, where

$$E(\alpha) := \{\delta_H(q_0, s) \in Q_H : \exists s \in \mathcal{L}(H) \text{ s.t. } P(s) = \alpha\}$$

is the current state estimate of α in H.

To deal with effect of control delay, $\text{Reach}^N(\cdot)$ operator is introduced to compute the set of all states that can be reached within N steps, i.e., for any $y \in 2^{Q_H}$, we have

$$\text{Reach}^N(y) = \{\delta_H(q, s) \in Q_H : \exists q \in y, s \in \Sigma^* \text{ s.t. } |s| \leq N\}.$$

The predictive supervisor, denoted by $S_{\text{pnc}} : \Sigma_o^* \to \Gamma$, is defined as follows. For any observable string $\alpha \in P(\mathcal{L}(H))$, the control decision of S_{pnc} is defined by

$$S_{\text{pnc}}(\alpha) = \Sigma \setminus \mathcal{D}(\text{Reach}^{N_c}(E(\alpha))) \tag{23}$$

where $\mathcal{D}(\cdot)$ represents the set of events that should be disabled as their occurrences may lead to illegal region, i.e., for any $y \in 2^{Q_H}$, we have

$$\mathcal{D}(y) = \{\sigma \in \Sigma_c : \exists q \in y \text{ s.t. } \delta(q, \sigma) \notin Q_{\text{good}}\}.$$

It is shown in [33] that the predictive supervisor is safe, i.e., $\mathcal{L}(S_{\text{pnc}}/G) \subseteq K$. This result is rather straightforward as it operates in a conservative manner by disabling all possible events that may cause illegal behavior.

Finally, for any observable string $\alpha \in P(\mathcal{L}(H))$, similar to the current-state estimate $E(\alpha)$, we also define $\hat{E}(\alpha)$ as the set of states that can be reached immediately after observing α without the unobservable tail, i.e.,

$$\hat{E}(\alpha) = \{\delta_H(s) \in Q_H : \exists s \in \Sigma^* \Sigma_o \cup \{\epsilon\} \text{ s.t. } P(s) = \alpha\}.$$

Clearly, we have $\hat{E}(\epsilon) = \{q_0\}$ and $E(\alpha) = UR(\hat{E}(\alpha))$.

B. Improving the Predictive Supervisor

As we can see, similar to the proposed online supervisor, the predictive supervisor S_{pnc} proposed in [33] also consists of the following two parts:

1) state estimation part: $\xi(y_0, \alpha) = E(\alpha)$, which estimates the current state of the system upon the occurrence of α using the observer of H;
2) state prediction part: $\text{Reach}^{N_c}(\cdot)$, which predicts states that can be reached in the next N_c steps.

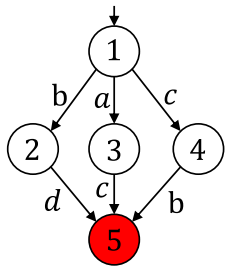

Fig. 3. System with $\Sigma = \Sigma_c = \Sigma_o$ and $N_c = 1$.

However, compared with our state estimation and prediction techniques, the state estimation and prediction techniques utilized in S_{pnc} are based on the *open-loop* systems. That is, both the estimation and the prediction are based on the original specification automaton H, which is independent from the control policy. This strategy is sufficient when considering the *supervisor existence problem* since we know *a priori* that the closed-loop language should be $K = \mathcal{L}(H)$. However, this is not the case for the *synthesis problem* since the closed-loop language is just a *sublanguage* of K in general. Therefore, using the original H to perform state estimation and state prediction may be overly conservative by taking some states that are not reachable in the closed-loop system into account.

The above observation suggests that the state estimation and state prediction techniques proposed in our article are more precise than those used in the predictive supervisor. Consequently, the proposed online supervisor should be strictly more permissive than the predictive supervisor. However, the following simple example shows that it is not the case if we choose control decision in lines 2 and 7 of Algorithm 1 arbitrarily and the proposed online supervisor and the predictive supervisor may be *incomparable*.

Example 5: Let us consider system G shown in Fig. 3 with $\Sigma = \Sigma_c = \Sigma_o$ and $N_c = 1$. The specification language K is generated by the subautomaton obtained by removing the single illegal state 5 from G. Let us discuss the initial control decision of the system.

First, using the predictive supervisor, we have $E(\epsilon) = \{1\}$, $Reach^{N_c}(\{1\}) = \{1, 2, 3, 4\}$ and $\mathcal{D}(\{1, 2, 3, 4\}) = \{b, c, d\}$. Therefore, the initial control decision is $S_{pnc}(\epsilon) = \{a\}$.

Now, let us use Algorithm 1 to compute the initial control decision of S_{online}. Initially, the algorithm starts from $\hat{\mathcal{E}}(\epsilon) = \{(q_0, \emptyset)\}$. Then consider the following two candidates of maximal control decision satisfying the condition in (19):

1) $S_{online}(\epsilon) = \{a, b\}$ or
2) $S_{online}(\epsilon) = \{c, d\}$.

In Algorithm 1, there is no specific criterion for which control decision we need to choose. However, if we choose $S_{online}(\epsilon) = \{c, d\}$, then the closed-loop language of the online supervisor will be incomparable with that of S_{pnc}.

The above example reveals the following issue. In general, there may have multiple maximal control decisions satisfying (19) in lines 2 and 7 of Algorithm 1. Specifically, some control decisions may not contain the control decision of S_{pnc} even though they are locally maximal. In order to guarantee that

the online supervisor can outperform the predictive supervisor, one straightforward way is to strengthen the condition in (19) by choosing control decision γ such that $S_{\text{pnc}}(\alpha) \subseteq \gamma$. Clearly, if the control decision γ chosen at each instant contains the control decision of the predictive supervisor, then S_{online} must outperform (at least equal to) S_{pnc}. Then the question arises as to whether or not we can *always* find such a "matching" control decision at each instant. The existence of such a control decision γ is not straightforward. In particular, in the partial observation setting, once a control decision of a supervisor is enlarged at some instant, the state estimate will become more uncertain in the future as more behaviors are allowed, which may, on the other hand, make the future control decision more conservative in order to maintain safety. Examples of such a phenomenon can be found in [5] and [43]. Therefore, the state prediction procedure should also take the future decisions of the given supervisor that we want to match into account in order to preserve the ability of "decision matching" at each instant. To this end, we generalize the USP to the *uncontrollable matched state prediction*, which not only takes all uncontrollable behaviors into account but also considers all possible behaviors we need to execute in order to match the given supervisor.

Definition 6: Let $\theta \in \Theta$ be a channel configuration, S be a supervisor, and $\alpha \in P(\mathcal{L}(S/G))$ be an observable string. Then the *uncontrollable matched language* from θ and α under supervisor S, denoted by $L^S_{uc}(\theta, \alpha)$, is defined recursively as follows.

1) $\epsilon \in L^S_{uc}(\theta, \alpha)$.
2) For any $s \in \Sigma^*, \sigma \in \Sigma$, we have $s\sigma \in L^S_{uc}(\theta, \alpha)$ if
   1) $|s| \leq N_c$;
   2) $s \in L^S_{uc}(\theta, \alpha)$;
   3) $\sigma \in \Gamma_{\geq |s|}(\theta) \cup (\bigcup_{i=0,1,\ldots,|s|} S(\alpha P(s_{-i})))$.

The intuition of the uncontrollable matched language is as follows. Suppose that S is a given supervisor, the channel configuration of the system is θ, and the current observation is α. Then all behaviors in $L^S_{uc}(\theta, \alpha)$ are unavoidable if we want to match the behavior under the control of S. Clearly, if we consider the most restrictive supervisor S_{res} that always disables all controllable events, then $L^{S_{\text{res}}}_{uc}(\theta, \alpha)$ coincides with $L_{uc}(\theta)$ for any α. Similar to the USP, we also define the *uncontrollable matched state prediction* as follows.

Definition 7: Let $\tilde{q} = (q, \theta) \in \tilde{Q}$ be an extended state, S be a supervisor, and $\alpha \in P(\mathcal{L}(S/G))$ be an observable string. Then the *uncontrollable matched state prediction* of $\tilde{q}$ w.r.t. S and α, denoted by $\text{USP}^S_M(\tilde{q}, \alpha)$, is defined by

$$\text{USP}^S_M(\tilde{q}, \alpha) = \{\delta(q, s) \in Q : s \in L^S_{uc}(\theta, \alpha)\}. \tag{24}$$

The uncontrollable matched state prediction is also extended to a set of extended states $x \in 2^{\tilde{Q}}$ by $\text{USP}^S_M(x, \alpha) = \cup_{\tilde{q} \in x} \text{USP}^S_M(\tilde{q}, \alpha)$.

Based on the above discussion, we propose a modified online control algorithm specified by Algorithm 2. The structures of Algorithms 1 and 2 are exactly the same. That is, upon each occurrence of new observable event, the supervisor chooses a new control decision by checking the USP assuming this decision is taken. The only difference is line 7, where the condition in (19) in Algorithm 1 is replaced by two conditions in Algorithm 2: (25) guarantees that the control decision choice at each instant contains the decision of the predictive supervisor and (26) guarantees that we should maintain the ability of "decision matching" in the future by looking ahead.

Algorithm 2: ONLINE-CONTROL-IP (G,N_c,Q_{good},S_{pnc}).

1 $\alpha \leftarrow \epsilon$, $\hat{\mathcal{E}}_S(\alpha) \leftarrow \{(q_0, \emptyset)\}$;
2 Find a maximal control decision $\gamma \in \Gamma$ such that

$$S_{pnc}(\alpha) \subseteq \gamma$$

and

$$\text{USP}^{S_{pnc}}_M(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha)), \alpha) \subseteq Q_{good}$$

3 Make control decision $S(\alpha) \leftarrow \gamma$;
4 $\mathcal{E}_S(\alpha) \leftarrow \text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))$;
while *a new event* $\sigma \in \Sigma_o$ *is observed* **do**
5 $\quad \hat{\mathcal{E}}_S(\alpha\sigma) \leftarrow \text{NOR}_\sigma(\mathcal{E}_S(\alpha))$;
6 $\quad \alpha \leftarrow \alpha\sigma$;
7 $\quad$ Find a maximal control decision $\gamma \in \Gamma$ such that

$$S_{pnc}(\alpha) \subseteq \gamma \tag{25}$$

$\quad$ and

$$\text{USP}^{S_{pnc}}_M(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha)), \alpha) \subseteq Q_{good} \tag{26}$$

8 $\quad$ Make control decision $S(\alpha) \leftarrow \gamma$;
9 $\quad \mathcal{E}_S(\alpha) \leftarrow \text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))$;

Next, we discuss the correctness of Algorithm 2. First, we show that Algorithm 2 is still recursively feasible, i.e., the modified conditions (25) and (26) can still be fulfilled recursively.

Theorem 4: Algorithm 2 is still recursively feasible. That is, at each instant of $\alpha \in P(\mathcal{L}(G))$, we can always recursively find a control decision $\gamma \in \Gamma$ such that both

1) $S_{\text{pnc}}(\alpha) \subseteq \gamma$ and
2) $\text{USP}^{S_{pnc}}_M(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha)), \alpha) \subseteq Q_{good}$.

Proof: See the Appendix.

Let us denote by S^*_{online} the modified online supervisor defined in Algorithm 2. According to the definitions of $\text{USP}(\cdot)$ and $\text{USP}^S_M(\cdot, \cdot)$, we have $\text{USP}(\tilde{q}) \subseteq \text{USP}^S_M(\tilde{q}, \alpha)$ for any S and α. This means that the choice of control decision in Algorithm 2 is more conservative than that of Algorithm 1. Hence, by the same argument in Theorem 3, we also know that S^*_{online} is safe, i.e., $\mathcal{L}(S^*_{\text{online}}/G) \subseteq K$. Furthermore, at each instant, $S^*_{\text{online}}(\alpha)$ is chosen such that $S_{\text{pnc}}(\alpha) \subseteq S^*_{\text{online}}(\alpha)$. Therefore, we also have $\mathcal{L}(S_{\text{pnc}}/G) \subseteq \mathcal{L}(S^*_{\text{online}}/G)$. The above discussion is summarized by the following theorem.

Theorem 5: Let G be a DES and S_{pnc} be the predictive supervisor and S^*_{online} be the modified online supervisor defined in Algorithm 2. Then we have $\mathcal{L}(S_{pnc}/G) \subseteq \mathcal{L}(S^*_{online}/G) \subseteq K$.

In [33], it was shown that controllability together with networked observability (the reader is referred to [33] for the definitions) provide the necessary and sufficient conditions under which K can be exactly achieved by the predictive supervisor. Then, we also have the following corollary.

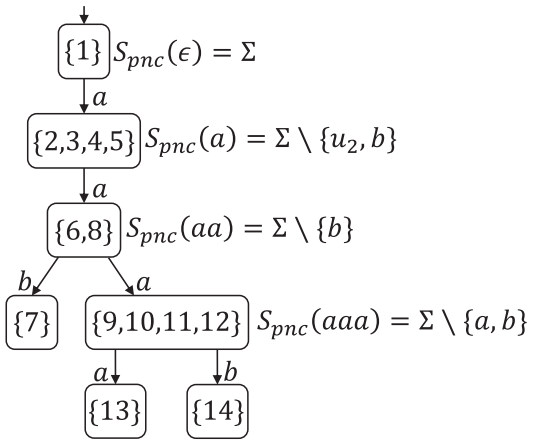

Fig. 4. Observer H_{obs} for system G in Fig. 2.

Corollary 2: Let G be a DES and S^*_{online} be the modified online supervisor. If K is controllable and networked observable, then we have $\mathcal{L}(S^*_{online}/G) = \mathcal{L}(S_{pnc}/G) = K$.

We now use the following example to demonstrate our online supervisor indeed works better than predictive supervisor.

Example 6: Again, we consider system G shown in Fig. 2 with $\Sigma_c = \Sigma, \Sigma_o = \{a, b\}$, and $N_c = 1$. We synthesize the predictive supervisor and the modified online supervisor to control the system, respectively, and compare their performances. First, we compute the predictive supervisor S_{pnc}. To this end, we need to compute the observer H_{obs}, which is shown in Fig. 4 . Then we use (23) to compute the control decision at each instant. For example, at the initial instant, we have

$$E(\epsilon) = \{1\}, Reach^{Nc}(E(\epsilon)) = \{1, 2\}$$

$$\text{and } S_{pnc}(\epsilon) = \Sigma \setminus \mathcal{D}(Reach^{Nc}(E(\epsilon))) = \Sigma.$$

When event a is observed, we then have

$$E(a) = \{2, 3, 4, 5\}, Reach^{Nc}(E(a)) = \{2, 3, 4, 5, 6, 8\}$$

$$\text{and } S_{pnc}(a) = \Sigma \setminus \mathcal{D}(Reach^{Nc}(E(a))) = \Sigma \setminus \{u_2, b\}.$$

Similarly, we can obtain $S_{pnc}(aa) = \Sigma \setminus \{b\}$ and $S_{pnc}(aaa) = \Sigma \setminus \{a, b\}$. The closed-loop language $\mathcal{L}(S_{pnc}/G)$ under control is generated by the automaton shown in Fig. 6(a).

Next, we synthesize the modified online supervisor S^*_{online}. Initially, we have $\hat{\mathcal{E}}_S(\epsilon) = \{(1, \emptyset)\}$. Since $S_{pnc}(\epsilon)$ is already Σ, by Theorem 4, we immediately get

$$S^*_{online}(\epsilon) = \Sigma =: \gamma_0$$

and update the extended state estimate to

$$\mathcal{E}_S(\epsilon) = \text{NUR}_{\gamma_0}(\hat{\mathcal{E}}_S(\epsilon)) = \{(1, \{(\gamma_0, 1)\})\}$$

which is the initial state of the structure shown in Fig. 5. When the first observable event a occurs, first, we update the extended state estimate to

$$\hat{\mathcal{E}}_S(a) = \text{NOR}_a(\mathcal{E}_S(\epsilon)) = \{(2, \{(\gamma_0, 0)\})\}.$$

Then we find a maximal control decision

$$S^*_{online}(a) = \Sigma \setminus \{u_2\} =: \gamma_1$$

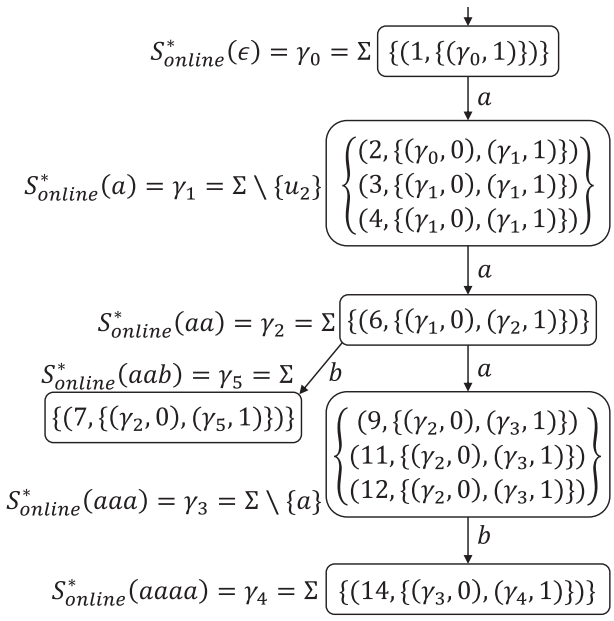

Fig. 5. Closed extended state estimate flow of G in Fig. 2.

such that $S_{pnc}(a) \subset \gamma_1$ and $\text{USP}_M^{S_{pnc}}(\text{NUR}_{\gamma_1}(\hat{\mathcal{E}}_S(a)), a) \subseteq Q_{good}$. To see the latter more clearly, we have

$$\text{NUR}_{\gamma_1}(\hat{\mathcal{E}}_S(a)) = \{(2, \theta), (3, \theta'), (4, \theta')\}$$

where $\theta = \{(\gamma_0, 0), (\gamma_1, 1)\}$ and $\theta' = \{(\gamma_1, 0), (\gamma_1, 1)\}$. Moreover, we can obtain the following:

$$L_{uc}^{S_{pnc}}(\theta, a) = \{\epsilon\} \cup (\gamma_0 \cup \gamma_1 \cup S_{pnc}(a)) \cup (\gamma_0 \cup \gamma_1 \cup S_{pnc}(a))\,(\gamma_1 \cup S_{pnc}(a))$$

$$L_{uc}^{S_{pnc}}(\theta', a) = \{\epsilon\} \cup (\gamma_1 \cup S_{pnc}(a)) \cup (\gamma_1 \cup S_{pnc}(a)) \,(\gamma_1 \cup S_{pnc}(a) \cup S_{pnc}(aa))$$

which gives $\text{USP}_M^{S_{pnc}}(\text{NUR}_{\gamma_1}(\hat{\mathcal{E}}_S(a)), a) = \{2, 3, 4, 6, 7, 9\} \subseteq Q_{good}$. Similarly, we can find $S^*_{online}(aa) = \Sigma =: \gamma_2$ as a control decision when string aa is observed. Note that at the instants of a and aa, S^*_{online} is strictly more permissive than S_{pnc}. Specifically, S_{pnc} needs to disable b to prevent the system from reaching illegal state 16 from state 8 via event b. However, by utilizing the control information, S^*_{online} knows that state 8 is not reachable as we have already disabled u_2. Therefore, the transition from state 6 to state 7, which does not exist in $\mathcal{L}(S_{pnc}/G)$, can be allowed by S^*_{online}. Finally, when string aaa is observed, S^*_{online} can choose either

1) $S^*_{online}(aaa) = \Sigma \setminus \{a\} := \gamma_3$
2) $S^*_{online}(aaa) = \Sigma \setminus \{b\} := \gamma'_3$.

Both of them strictly contain the corresponding control decision $S_{pnc}(aaa) = \Sigma \setminus \{a, b\}$ issued by the predictive supervisor. Suppose that we choose $S^*_{online}(aaa) = \gamma_3$. Then the entire information flow of the extended state estimate under S^*_{online} and the associated control decisions are depicted in Fig. 5. The closed-loop language $\mathcal{L}(S^*_{online}/G)$ under control is generated by the automaton shown in Fig. 6(b). Clearly, we see that the

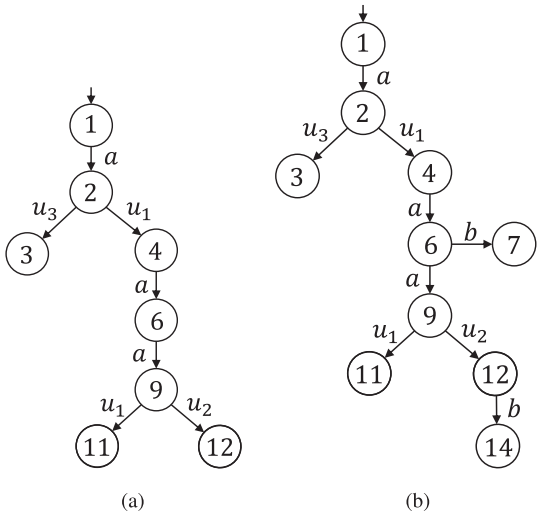

Fig. 6. Closed-loop languages induced by S_{pnc} and S_{online}, respectively. (a) $\mathcal{L}(S_{pnc}/G)$. (b) $\mathcal{L}(S^*_{online}/G)$.

modified online supervisor S^*_{online} is strictly more permissive than the predictive supervisor S_{pnc}.

Remark 5: We conclude by discussing the complexity of the proposed online control algorithms. First, operator $\text{NX}(\theta)$ is just a one-step reachability operation, which is linear in the size of θ ($|\theta| \leq N_c$). Also, the complexities of computing $\text{NOR}_\sigma(x)$ and $\text{NUR}_\gamma(x), x \in 2^{\tilde{Q}}$ are $O(|Q| \cdot |\Theta|)$ and $O(|\Sigma| \cdot |Q| \cdot |\Theta|)$, respectively. Furthermore, the complexity for computing $\text{USP}(x), x \in 2^{\tilde{Q}}$ is also $O(|\Sigma| \cdot |Q| \cdot |\Theta|)$, which is essentially an N_c-step reachability search. Therefore, in Algorithm 1, it takes $O(|\Sigma| \cdot |Q| \cdot |\Theta|)$ to compute $\hat{\mathcal{E}}_S(\cdot), \mathcal{E}_S(\cdot)$ (lines 1, 4, 5, and 9) and takes $O(|\Gamma| \cdot |\Sigma| \cdot |Q| \cdot |\Theta|), |\Gamma| = 2^{|\Sigma_c|}$ to find some maximal control decision γ (lines 2 and 7) because we need to test all possible control decisions in Γ and for each control decision to compute its USP and test the set inclusion (which can be done simultaneously).

The case of Algorithm 2 is similar, the only difference is the computation of uncontrollable matched state prediction $\text{USP}^{S_{pnc}}_M(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha)), \alpha)$. Here, we also need to use additionally computed control decisions of the predictive supervisor. Note that the predictive supervisor can be implemented by a networked observer that is exponential in the size of G. However, this part can be done *offline* and be stored. Therefore, the online computation of $\text{USP}^{S_{pnc}}_M(\cdot)$ is still $O(|\Sigma| \cdot |Q| \cdot |\Theta|)$ and the overall complexity of Algorithm 2 for each online instant (each execution of the while-loop) is also $O(|\Gamma| \cdot |\Sigma| \cdot |Q| \cdot |\Theta|)$.

Finally, we note that the set Θ contains no more than $|\Gamma|^{N_c} = 2^{|\Sigma_c| \cdot N_c}$ channel configurations. Therefore, the complexity of the online algorithm for each execution is polynomial in the number of states but exponential in the number of controllable events and the delay bound. In practice, the number of events is much smaller than the number of states in G. Furthermore, in most real-world applications, N_c is usually relatively small. Otherwise, it is more natural for engineers to first improve the hardware to reduce delays and then to use networked control algorithms to compensate the effect of delays.

VI. Conclusion

We investigated the supervisor synthesis problem in the context of networked DESs. First, a novel structure called channel configuration was proposed to dynamically capture past control history; hence, closed-loop state estimate was realized and it was more accurate compared with that computed according to the observer built beforehand. Then, based on the channel configuration, we proposed an operator called USP to guarantee that the system always retains in the safe region under our online supervisory control and put forward a safe and recursive feasible algorithm. Finally, we formally prove that our proposed online supervisor always works better than the predictive supervisor in the sense that it can always obtain a larger (at least equal to) event set than the latter along the system trajectory. Note that the control specification considered in this work is safety; liveness specifications such as nonblockingness and deadlock-freeness are not considered. The main reason is that liveness is a global requirement that cannot be enforced using online control algorithm on-the-fly. To guarantee liveness, one may need to first construct the entire reachable information space offline and then compute the supervisor, which loses the main computational advantage of the online control approach. Also, this work only considers communication delays in control channel. In the future, we also plan to extend our results to DES with both observation and control delays.

Appendix

Proof of Theorem 4

Proof: To show this, it suffices to show that

$$\text{USP}^{S_{\text{pnc}}}_M(\text{NUR}_{S_{\text{pnc}}(\alpha)}(\hat{\mathcal{E}}_S(\alpha)), \alpha) \subseteq Q_{\text{good}}.$$

That is, at least choosing the decision of S_{pnc} satisfies (25) and (26). We proceed by induction on the length of the observation string.

Induction Basis: Initially, we have $\alpha = \epsilon$ and $\hat{\mathcal{E}}_S(\epsilon) = \{(q_0, \emptyset)\}$. Let (q, θ) be an arbitrary extended state in $\text{NUR}_{S_{\text{pnc}}(\epsilon)}(\hat{\mathcal{E}}_S(\epsilon))$ and $s \in (S_{\text{pnc}}(\epsilon) \cap \Sigma_{uo})^*$ be an arbitrary string such that $(\delta(s), \theta(s)) = (q, \theta)$. First, we know that $q \in Q_{\text{good}}$ as S_{pnc} is a safe supervisor. Then we consider an arbitrary string $w \in L^{S_{\text{pnc}}}_{uc}(\theta, \epsilon)$. By matching the definition of $\mathcal{L}(S/G)$ and $L^{S_{\text{pnc}}}_{uc}(\theta, \epsilon)$, we have $sw \in \mathcal{L}(S_{\text{pnc}}/G)$. This implies that $\delta(q_0, sw) \in Q_{\text{good}}$. Since all strings and states considered are arbitrary, we have

$$\text{USP}^{S_{\text{pnc}}}_M(\text{NUR}_{S_{\text{pnc}}(\epsilon)}(\hat{\mathcal{E}}_S(\epsilon)), \epsilon) \subseteq Q_{\text{good}}.$$

This proves the induction basis.

Induction Step: Now, let $\alpha\sigma \in \Sigma^*_o$ be an observable sequence that can happen under the control of the modified online supervisor, where $\sigma \in \Sigma_o$ and $|\alpha| = k$. We assume that (25) and (26) can always be fulfilled at each instant along α and we need to show that control decision $S_{\text{pnc}}(\alpha\sigma)$ can still fulfill these conditions for the instant of $\alpha\sigma$.

Let us consider an arbitrary extended state

$$(q, \theta) \in \text{NUR}_{S_{\text{pnc}}(\alpha\sigma)}(\text{NOR}_\sigma(\mathcal{E}_S(\alpha))).$$

By the definitions of NUR and NOR, there exists a state $(\hat{q},\hat{\theta}) \in \mathcal{E}_S(\alpha)$ and a string $\sigma_1\sigma_2\ldots\sigma_m \in \Sigma_{uo}^*$ that induce the following sequence of extended states:

$$(\hat{q},\hat{\theta}) \xrightarrow{\sigma} (q_1,\theta_1) \xrightarrow{\sigma_1} \cdots \xrightarrow{\sigma_m} (q_{m+1},\theta_{m+1})$$

where we have the following:

1) $\sigma \in \Gamma(\hat{\theta})$ and $(q_1,\theta_1) = (\delta(\hat{q},\sigma), \mathrm{NX}(\hat{\theta}) \cup \{(S_{\mathrm{pnc}}(\alpha\sigma), N_c)\})$;
2) for any $1 \le i \le m$, we have $\sigma_i \in \Gamma(\theta_i)$ and $(q_{i+1},\theta_{i+1}) = (\delta(q_i,\sigma_i), \mathrm{NX}(\theta_i) \cup \{(S_{\mathrm{pnc}}(\alpha\sigma), N_c)\})$;
3) $(q_{m+1},\theta_{m+1}) = (q,\theta)$.

Since $(\hat{q},\hat{\theta}) \in \mathcal{E}_S(\alpha)$, we know that there exists a string $s \in \mathcal{L}(S/G)$ such that $P(s) = \alpha$ and $(\delta(s),\theta(s)) = (\hat{q},\hat{\theta})$. Note that, although supervisor S has only made control decisions up to the instant of α, $\mathcal{L}(S/G)$ and $\theta(s)$ are still well defined as they only depend on the information in the history. Moreover, by the induction hypothesis, we have

$$\mathrm{USP}_M^{S_{\mathrm{pnc}}}((\delta(s),\theta(s)),\alpha) \subseteq Q_{\mathrm{good}}. \tag{27}$$

Now, let us consider an arbitrary string $w \in L_{uc}^{S_{\mathrm{pnc}}}(\theta,\alpha\sigma)$ such that $\delta(q,w)$! To be more specific, we write w in the form of $w = \sigma_w^1\sigma_w^2\ldots\sigma_w^{|w|} \in \Sigma^*$. By Definition 6, we have $|w| \le N_c + 1$ and

$$\sigma_w^{i+1} \in \Gamma_{\ge i}(\theta_{m+1}) \cup \left(\bigcup_{k=0,\ldots,i} S(\alpha\sigma P((\sigma_w^1\ldots\sigma_w^i)_{-k})) \right).$$

Then we consider the following two cases.

Case 1: $1 + m + |w| \le N_c + 1$.

In such case, by considering $L_{uc}^{S_{\mathrm{pnc}}}(\hat{\theta},\alpha)$ and the above specified sequence of extended states and string w, we have $\sigma\sigma_1\cdots\sigma_m w \in L_{uc}^{S_{\mathrm{pnc}}}(\hat{\theta},\alpha)$. Therefore, we know that $\delta(\hat{q},\sigma\sigma_1\cdots\sigma_m w) = \delta(q,w) \in Q_{\mathrm{good}}$ by (27).

Case 2: $1 + m + |w| > N_c + 1$.

In such case, we proceed our proof by two parts. Without loss of generality, we assume that $m \ge N_c$. If $m < N_c$ holds, we will see that, on the basis of Case 1, the proof is degenerated to only the second part.

First, we prove that the state reached via unobservable string is in Q_{good}. By Corollary 1 and the definitions of $E(\alpha)$ and $\hat{E}(\alpha)$, we know that $\hat{q} \in Q(\mathcal{E}_S(\alpha)) \subseteq E(\alpha)$, which implies $q_1 \in Q(\hat{\mathcal{E}}_S(\alpha\sigma)) \subseteq \hat{E}(\alpha\sigma)$. Note that S_{pnc} in (23) can also be rewritten as

$$\begin{aligned} S_{\mathrm{pnc}}(\alpha\sigma) &= \Sigma \setminus \mathcal{D}(\mathrm{Reach}^{N_c}(E(\alpha\sigma))) \\ &= \Sigma \setminus \mathcal{D}(\mathrm{Reach}^{N_c}(\mathrm{UR}(\hat{E}(\alpha\sigma)))) \end{aligned}$$

which means no event in $S_{\mathrm{pnc}}(\alpha\sigma)$ can lead a state in $\mathrm{Reach}^{N_c}(\mathrm{UR}(\hat{E}(\alpha\sigma)))$ out of Q_{good}. Furthermore, since q_{m+1} is reached under $\mathrm{NUR}_{S_{\mathrm{pnc}}(\alpha\sigma)}(\cdot)$ operator, and $\sigma_1\cdots\sigma_m \in \Sigma_{uo}^*$ with $\sigma_i \in \Gamma(\theta_i)$, following the same argument in Case 1, i.e., no event in $\Gamma(\hat{\theta})$ can lead a state in $\mathrm{Reach}^{N_c}(\mathrm{UR}(\hat{E}(\alpha\sigma)))$ out of Q_{good}, we know that $q_1, q_2, \ldots, q_{m+1} \in Q_{\mathrm{good}}$.

Second, we prove that the state reached via $w \in L_{uc}^{S_{\mathrm{pnc}}}(\theta_{m+1},\alpha\sigma)$ is also in Q_{good}. For this, let us consider states reached along string w. Note that $\Gamma(\theta_{m+1}) = S_{\mathrm{pnc}}(\alpha\sigma)$ since $m \ge N_c$. Then consider the first state $q^1 = \delta(q_{m+1},\sigma_w^1)$. If $\sigma_w^1 \in \Sigma_{uo}$, then this reduces to the first part. Therefore, we assume $\sigma_w^1 \in \Sigma_o$ and a new control decision $S(\alpha\sigma P(\sigma_w^1))$ will be issued. Since q^1 is reached from state $q_{m+1} \in \mathrm{Reach}^{N_c}(E(\alpha\sigma))$ via event $\sigma_w^1 \in S_{\mathrm{pnc}}(\alpha\sigma)$, we know from the same argument in the first part that $q^1 \in Q_{\mathrm{good}}$. Then consider the second state $q^2 = \delta(q_{m+1},\sigma_w^1\sigma_w^2)$, where we have $\sigma_w^2 \in S_{\mathrm{pnc}}(\alpha\sigma) \cup S_{\mathrm{pnc}}(\alpha\sigma\sigma_w^1)$. On the one hand, if $\sigma_w^2 \in S_{\mathrm{pnc}}(\alpha\sigma)$, then we obtain immediately that $q^2 \in Q_{\mathrm{good}}$ as $q^1 \in \mathrm{Reach}^{Nc}(E(\alpha\sigma))$. On the other hand, if $\sigma_w^2 \in S_{\mathrm{pnc}}(\alpha\sigma\sigma_w^1)$, then based on the construction of $S_{\mathrm{pnc}}(\alpha\sigma\sigma_w^1)$, i.e., $\Sigma \setminus \mathcal{D}(\mathrm{Reach}^{Nc}(E(\alpha\sigma\sigma_w^1)))$, we also have $q^2 \in Q_{\mathrm{good}}$ since we know for sure that $q^1 \in \mathrm{Reach}^{N_c}(E(\alpha\sigma\sigma_w^1))$. This means that for any feasible event, no matter it is enabled by past control decisions in the channel or by the latest effective control decision, the transition from state q^1 to q^2 is always safe as it is also allowed in the predictive supervisor. Therefore, by applying the above argument iteratively and the fact that $|w| \le N_c + 1$, we have $q^{|w|} \in Q_{\mathrm{good}}$, where $q^{|w|} = \delta(q_{m+1},\sigma_w^1\cdots\sigma_w^{|w|})$.

Because w is an arbitrary string in $L_{uc}^{S_{\mathrm{pnc}}}(\theta,\alpha\sigma)$, we know that $\mathrm{USP}_M^{S_{\mathrm{pnc}}}(q,\theta) \subseteq Q_{\mathrm{good}}$. Since (q,θ) is also chosen arbitrarily, we further have

$$\mathrm{USP}_M^{S_{\mathrm{pnc}}}(\mathrm{NUR}_{S_{\mathrm{pnc}}(\alpha\sigma)}(\mathrm{NOR}_\sigma(\mathcal{E}_S(\alpha))),\alpha\sigma) \subseteq Q_{\mathrm{good}}$$

which completes the induction step.

REFERENCES

[1] M. V. S. Alves and J. C. Basilio, "State estimation and detectability of networked discrete event systems with multi-channel communication networks," in *Proc. IEEE Amer. Control Conf.*, 2019, pp. 5602–5607.

[2] M. V. S. Alves, J. C. Basilio, A. E. Carrilho da Cunha, L. K. Carvalho, and M. V. Moreira, "Robust supervisory control against intermittent loss of observations," in *Proc. 12th Int. Workshop Discrete Event Syst.*, vol. 12, 2014, 12, pp. 294–299.

[3] M. V. S. Alves, L. K. Carvalho, and J. C. Basilio, "Supervisory control of timed networked discrete event systems," in *Proc. 56th IEEE Conf. Decis. Control*, 2017, pp. 4859–4865.

[4] S. Balemi, "Input/output discrete event processes and communication delays," *Discrete Event Dyn. Syst.*, vol. 4, no. 1, pp. 41–85, 1994.

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

[6] K. Cai, R. Zhang, and W. M. Wonham, "Relative observability of discrete-event systems and its supremal sublanguages," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 659–670, Mar. 2015.

[7] L. K. Carvalho, J. C. Basilio, and M. V. Moreira, "Robust diagnosis of discrete event systems against intermittent loss of observations," *Automatica*, vol. 48, no. 9, pp. 2068–2078, 2012.

[8] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. Berlin, Germany: Springer, 2008.

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

[10] C. Gu, X. Wang, and Z. Li, "Synthesis of supervisory control with partial observation on normal state-tree structures," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 984–997, Apr. 2019.

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

[12] R. Jacob, J.-J. Lesage, and J.-M. Faure, "Overview of discrete event systems opacity: Models, validation, and quantification," *Annu. Rev. Control*, vol. 41, pp. 135–146, 2016.

[13] J. Komenda and F. Lin, "Modular supervisory control of networked discrete-event systems," in *Proc. 13th Int. Workshop Discrete Event Syst.*, 2016, pp. 85–90.
[14] J. Komenda, T. Masopust, and J. H. Van Schuppen, "Synthesis of controllable and normal sublanguages for discrete-event systems using a coordinator," *Syst. Control Lett.*, vol. 60, no. 7, pp. 492–502, 2011.
[15] J. Komenda, T. Masopust, and J. H. van Schuppen, "Supervisory control synthesis of discrete-event systems using a coordination scheme," *Automatica*, vol. 48, no. 2, pp. 247–254, 2012.
[16] F. Lin, "Control of networked discrete event systems: Dealing with communication delays and losses," *SIAM J. Control Optim.*, vol. 52, no. 2, pp. 1276–1298, 2014.
[17] Z. Liu, X. Yin, S. Shu, and S. Li, "Online supervisory control of networked discrete-event systems with control delays," in *Proc. 58th IEEE Conf. Decis. Control*, 2019, pp. 6706–6711.
[18] J. Lunze, *Control Theory of Digitally Networked Dynamic Systems*. Berlin, Germany: Springer, 2014.
[19] C. E. V. Nunes, M. V. Moreira, M. V. S. Alves, L. K. Carvalho, and J. C. Basilio, "Codiagnosability of networked discrete event systems subject to communication delays and intermittent loss of observation," *Discrete Event Dyn. Syst.*, vol. 28, no. 2, pp. 215–246, Jun. 2018.
[20] S.-J. Park and K.-H. Cho, "Delay-robust supervisory control of discrete-event systems with bounded communication delays," *IEEE Trans. Autom. Control*, vol. 51, no. 5, pp. 911–915, May 2006.
[21] S.-J. Park and K.-H. Cho, "Supervisory control of discrete event systems with communication delays and partial observations," *Syst. Control Lett.*, vol. 56, no. 2, pp. 106–112, 2007.
[22] S.-J. Park and K.-H. Cho, "Achieving a global objective with competing networked agents in the framework of discrete event systems," *Int. J. Control*, vol. 93, no. 4, pp. 889–897, Apr. 2020.
[23] S. Pruekprasert and T. Ushio, "Supervisory control of communicating timed discrete event systems for state avoidance problem," *IEEE Contr. Syst. Lett.*, vol. 4, no. 1, pp. 259–264, Jan. 2020.
[24] A. Rashidinejad, M. Reniers, and M. Fabian, "Supervisory control of discrete-event systems in an asynchronous setting," in *Proc. 15th Int. Conf. Autom. Sci. Eng.*, 2019, pp. 494–501.
[25] A. Rashidinejad, M. Reniers, and L. Feng, "Supervisory control of timed discrete-event systems subject to communication delays and non-fifo observations," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 456–463.
[26] K. Rohloff, "Sensor failure tolerant supervisory control," in *Proc. 44th IEEE Conf. Decis. Control*, 2005, pp. 3493–3498.
[27] W. H. Sadid, L. Ricker, and S. Hashtrudi-Zad, "Robustness of synchronous communication protocols with delay for decentralized discrete-event control," *Discrete Event Dyn. Syst.*, vol. 25, no. 1-2, pp. 159–176, 2015.
[28] A. M. Sánchez and F. J. Montoya, "Safe supervisory control under observability failure," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 16, no. 4, pp. 493–525, 2006.
[29] Y. Sasi and F. Lin, "Detectability of networked discrete event systems," *Discrete Event Dyn. Syst.*, vol. 28, no. 3, pp. 449–470, 2018.
[30] S. Shu and F. Lin, "Decentralized control of networked discrete event systems with communication delays," *Automatica*, vol. 50, no. 8, pp. 2108–2112, 2014.
[31] S. Shu and F. Lin, "Supervisor synthesis for networked discrete event systems with communication delays," *IEEE Trans. Autom. Control*, vol. 60, no. 8, pp. 2183–2188, Aug. 2015.
[32] S. Shu and F. Lin, "Deterministic networked control of discrete event systems with nondeterministic communication delays," *IEEE Trans. Autom. Control*, vol. 62, no. 1, pp. 190–205, Jan. 2017.
[33] S. Shu and F. Lin, "Predictive networked control of discrete event systems," *IEEE Trans. Autom. Control*, vol. 62, no. 9, pp. 4698–4705, Sep. 2017.
[34] S. Takai and T. Ushio, "Effective computation of an $L_m(G)$-closed, controllable, and observable sublanguage arising in supervisory control," *Syst. Control Lett.*, vol. 49, no. 3, pp. 191–200, 2003.
[35] S. Takai and T. Ushio, "Verification of codiagnosability for discrete event systems modeled by mealy automata with nondeterministic output functions," *IEEE Trans. Autom. Control*, vol. 57, no. 3, pp. 798–804, Mar. 2012.
[36] T. Ushio and S. Takai, "Nonblocking supervisory control of discrete event systems modeled by mealy automata with nondeterministic output functions," *IEEE Trans. Autom. Contr*ol, vol. 61, no. 3, pp. 799–804, Mar. 2016.
[37] G. S. Viana, M. S. Alves, and J. C. Basilio, "Codiagnosability of timed networked discrete-event systems subject to event communication delays and intermittent loss of observation," in *Proc. 56th IEEE Conf. Decis. Control*, 2017, pp. 4211–4216.
[38] W. M. Wonham and K. Cai, *Supervisory Control of Discrete-Event Systems*. Berlin, Germany: Springer, 2018.
[39] S. Yang, J. Hou, X. Yin, and S. Li, "Opacity of networked supervisory control systems over insecure communication channels," *IEEE Control Netw. Syst.*, to be published, 2021, doi: 10.1109/TCNS.2021.3050131.
[40] X. Yin, "Supervisor synthesis for mealy automata with output functions: A model transformation approach," *IEEE Trans. Autom. Control*, vol. 62, no. 5, pp. 2576–2581, May 2017.
[41] X. Yin and S. Lafortune, "Synthesis of maximally permissive supervisors for partially observed discrete event systems," *IEEE Trans. Autom. Control*, vol. 61, no. 5, pp. 1239–1254, May 2016.
[42] X. Yin and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems," *IEEE Trans. Autom. Control*, vol. 61, no. 8, pp. 2140–2154, Aug. 2016.
[43] X. Yin and S. Lafortune, "Synthesis of maximally-permissive supervisors for the range control problem," *IEEE Trans. Autom. Control* vol. 62, no. 8, pp. 3914–3929, Aug. 2017.
[44] M. Zgorzelski and J. Lunze, "A method for the synchronisation of networked discrete-event systems," in *Proc. 13th Int. Workshop Discrete Event Syst.*, 2016, pp. 444–451.
[45] M. Zgorzelski and J. Lunze, "A. new approach to tracking control of networked discrete-event systems," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 448–455.
[46] M. Zgorzelski, P. Ostendarp, and J. Lunze, "Experimental application of a networked tracking control method to the discrete-event demonstrator HANS," in *Proc. IEEE Conf. Control Tech. Appl.*, 2018, pp. 450–457.
[47] R. Zhang, K. Cai, Y. Gan, and W. M. Wonham, "Delay-robustness in distributed control of timed discrete-event systems based on supervisor localisation," *Int. J. Control*, vol. 89, no. 10, pp. 2055–2072, 2016.
[48] R. Zhang, K. Cai, Y. Gan, and W. M. Wonham, "Distributed supervisory control of discrete-event systems with communication delay," *Discrete Event Dyn. Syst.*, vol. 26, no. 2, pp. 263–293, 2016.
[49] B. Zhao, F. Lin, C. Wang, X. Zhang, M. P. Polis, and L. Y. Wang, "Supervisory control of networked timed discrete event systems and its applications to power distribution networks," *IEEE Control Netw. Syst.*, vol. 4, no. 2, pp. 146–158, Jun. 2017.
[50] Y. Zhu, L. Lin, S. Ware, and R. Su, "Supervisor synthesis for networked discrete event systems with communication delays and lossy channels," in *Proc. 58th IEEE Conf. Decis. Control*, 2019, pp. 6730–6735.

Zhaocong Liu (Student Member, IEEE) received the B.Eng degree in mechanical engineering from Zhejiang University in 2018, the M.S. degree in control engineering from the Shanghai Jiaotong University, in 2021. He is currently working toward the Ph.D. degree at the Chinese University of Hong Kong.

His research interests includes the supervisory control of discrete event systems, multi-agent systems andnonlinear control systems.

Mr. Liu was a recipient of the National Scholarship from China.

Xiang Yin (Member, IEEE) was born in Anhui, China, in 1991. He received the B.Eng. degree from Zhejiang University, Hangzhou, China, in 2012, and the M.S. and Ph.D. degrees from the University of Michigan, Ann Arbor, MI, USA, in 2013 and 2017, respectively, all in electrical engineering.

Since 2017, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is currently an Associate Professor. His research interests include formal methods, discrete-event systems, and cyber–physical systems.

Dr. Yin is serving as the Co-Chair of the IEEE CSS Technical Committee on Discrete Event Systems, an Associate Editor for the *Journal of Discrete Event Dynamic Systems: Theory & Applications*, and a member of the IEEE CSS Conference Editorial Board. He was the recipient of the Best Student Paper Award Finalist in IEEE Conference on Decision and Control (CDC) 2016.

Shaolong Shu (Senior Member, IEEE) received the B.Eng. degree in automatic control and the Ph.D. degree in control theory and control engineering from Tongji University, Shanghai, China, in 2003 and 2008, respectively.

Since 2008, he has been with the School of Electronics and Information Engineering, Tongji University, where he is currently a Full Professor. From 2007 to 2008 and from 2014 to 2015, he was a Visiting Scholar with Wayne State University, Detroit, MI, USA. His main research interests include state estimation and control of discrete event systems and cyber–physical systems.

Feng Lin (Fellow, IEEE) received the B.Eng. degree in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 1982, and the M.A.Sc. and Ph.D. degrees in electrical engineering from the University of Toronto, Toronto, ON, Canada, in 1984 and 1988, respectively.

He was a Postdoctoral Fellow with Harvard University, Cambridge, MA, USA, from 1987 to 1988. Since 1988, he has been with the Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI, USA, where he is currently a Professor. His current research interests include discrete event systems, hybrid systems, robust control, and their applications in alternative energy, biomedical systems, and automotive control. He has authored a book entitled *Robust Control Design: An Optimal Control Approach.*

Dr. Lin coauthored a paper that received a George Axelby Outstanding Paper Award from the IEEE Control Systems Society. He was an Associate Editor for IEEE TRANSACTIONS ON AUTOMATIC CONTROL.

Shaoyuan Li (Senior Member, IEEE) was born in Hebei, China, in 1965. He received the B.S. and M.S. degrees in automation from the Hebei University of Technology, Tianjin, China, in 1987 and 1992, respectively, and the Ph.D. degree from Nankai University, Tianjin, China, in 1997.

Since 1997, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is currently a Professor. His current research interests include model predictive control, dynamic system optimization, and cyber–physical systems.

Dr. Li is the Vice-President of the Chinese Association of Automation.