2019 IEEE 58th Conference on Decision and Control (CDC)
Palais des Congrès et des Expositions Nice Acropolis
Nice, France, December 11-13, 2019

Online Supervisory Control of Networked Discrete-Event Systems with Control Delays

Zhaocong Liu, Xiang Yin, Shaolong Shu and Shaoyuan Li

Abstract— **We investigate state estimation and safe controller synthesis for networked discrete-event systems (DES), 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 paper, 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. 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.**

I. Introduction

In this paper, 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. In many modern applications, supervisors are connected to the plants via communication channels. Although such networked information structures provide more flexible ways for controlling DES, it also brings new research challenges. Therefore, networked DES has drawn many attention in the past few years in the DES literature; see, e.g., [1]–[5].

The study of supervisory control of networked DES dates back to the work of Balemi [6], where the robustness of supervisors under communication delays was investigated. In [7], Park and Cho also studied the supervisor control problem of networked DES assuming that each controllable event is disabled by default. More recently, Lin [8] 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 network controllability and network observability, were provided for the existence of a non-predictive 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 [9]–[11]. In [11], 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 extended to modular systems by [10]. In [12], [13], supervisory control of networked DES has also been extended to the timed setting.

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 sub-language 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 [14]–[18]. In the context of control of networked DES, in [19]–[21], supervisor synthesis under observation delays and losses were investigated. To our knowledge, however, the supervisor synthesis problem with control delays has not yet been fully investigated in the literature. One exception is the work of [9], where the supervisor synthesis problem for safety specification under both control and observation delays is tackled. However, the approach in [9] requires to restrict the solution space a priori, while the solution space is infinite in general.

In this paper, we investigate the supervisor synthesis problem for networked discrete event systems with safety specifications. Specifically, we focus on the case of control delays and assume that the system is only partially-observed. The synthesis problem under control delays is quite different from the case of observation delays. In particular, due to control delays, the control decision issued currently may affect the closed-loop behavior in the future; this issue has to be taken into account in the synthesis problem. In this paper, we proposed a novel online control algorithm for networked DES with control delays. First, we propose a new approach for estimating the state of the closed-loop system recursively online. Then, we discuss how to choose a control decision online by predicting its effect in the future. Finally, we integrate the state estimation and the state prediction 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., [15], [22]. To the best of our knowledge, online supervisory control has never been applied to networked DES with communication delays.

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

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

Z. Liu, X. Yin and S. Li are with Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: {zhaocongl,yinxiang,syli}@sjtu.edu.cn.

S. Shu is with the School of Electronics and Information Engineering, Tongji University, Shanghai 201804, China. E-mail: shushaolong@tongji.edu.cn.

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 $s = \sigma_1 \sigma_2 \cdots \sigma_n, \sigma_i \in \Sigma$, 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., $s_{-i} = \sigma_1 \ldots \sigma_{|s|-i}$; 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 discrete event system 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 usually manner; see, e.g., [23]. 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 $\mathcal{L}(G) = \{s \in \Sigma^* : \delta(s)!\}$.

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.

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 in the usual manner; see, e.g., [23]. That is, upon the occurrence of a string $s \in \Sigma^*$, the supervisor can only observed $P(s) \in \Sigma_o^*$. 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 non-negative integer N_c. Then the language generated by the closed-loop system (subject to control delays), denoted by $\mathcal{L}(S/G)$, is defined recursively as follows [8]:

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

Remark 1: 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 [24]. This definition is very suitable for the purpose of safety as it is conservative by capturing all possible actions the supervisor may take under control delays.

Our goal is to design a supervisor such that the closed-loop system satisfies a *safety specification*. 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 paper as follows.

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 sub-automaton* of G, denoted by $H = (Q_H, \Sigma, \delta_H, q_0)$, where $Q_H \subseteq Q$ and $\delta_H \subseteq \delta$. That is, $\mathcal{L}(H) = K \subseteq \mathcal{L}(G)$ 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

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

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

Remark 2: In [11], the authors investigated under what condition the specification language K can be *exactly* achieved subject to control delays. This problem is referred to as the *supervisor existence problem* in the literature [23]. 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*.

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 $\mathcal{L}(S/G)$ [8]. However, this state estimation technique essentially utilize 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:

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

where each $\gamma_i \in \Gamma$ is a control decision and each $n_i \in [0, N_c]$ is a non-negative 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 define $\Theta := 2^{\Gamma \times [0,N_c]}$ as the set of all channel configurations.

Intuitively, each channel configuration specifies the control decisions 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.

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

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

That is, $\mathrm{NX}(\theta)$ decreases the timing index of each control decision in θ one unit and it only keeps control decisions whose timing indices are non-negative (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

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

We formalize the estimation updating procedures for the above two scenarios by operators $\mathrm{NUR} : 2^{\tilde{Q}} \times \Gamma \to 2^{\tilde{Q}}$ and $\mathrm{NOR} : 2^{\tilde{Q}} \times \Sigma_o \to 2^{\tilde{Q}}$, 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 *networked unobservable reach* (NUR) of x under γ, denoted by $\mathrm{NUR}_\gamma(x)$, is defined recursively as follows:

- For any extended state $(q, \theta) \in x$, we have
$$(q, \theta \cup \{(\gamma, N_c)\}) \in \mathrm{NUR}_\gamma(x) \tag{1}$$
- For any extended state $(q, \theta) \in \mathrm{NUR}_\gamma(x)$ and any unobservable event $\sigma \in \Sigma_{uo}$, we have
$$\begin{aligned} &\sigma \in \Gamma(\theta) \text{ and } \delta(q, \sigma)! \\ \Rightarrow &(\delta(q, \sigma), \mathrm{NX}(\theta) \cup \{(\gamma, N_c)\}) \in \mathrm{NUR}_\gamma(x). \end{aligned} \tag{2}$$

Intuitively, $\mathrm{NUR}_\gamma(x)$ 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 $(q, \theta) \in \mathrm{NUR}_\gamma(x)$, 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 $(\delta(q, \sigma), \mathrm{NX}(\theta) \cup \{(\gamma, N_c)\})$ to $\mathrm{NUR}_\gamma(x)$. The second component of the successor state is $\mathrm{NX}(\theta) \cup \{(\gamma, N_c)\}$ since (i) the execution of σ will decrease the timing index of each previous control decision in θ by one unit; and (ii) we 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 *networked observable reach* (NOR) of x upon the occurrence of σ, denoted by $\mathrm{NOR}_\sigma(x)$, is defined by

$$\mathrm{NOR}_\sigma(x) = \{(\delta(q, \sigma), \mathrm{NX}(\theta)) \in \tilde{Q} : (q, \theta) \in x, \sigma \in \Gamma(\theta)\}. \tag{3}$$

Intuitively, $\mathrm{NOR}_\sigma(x)$ 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.

Example 1: Let us consider system G shown in Figure 1 with $\Sigma_o = \{a\}, \Sigma_c = \Sigma$ and $N_c = 1$. Suppose that the current extended state estimate is $x_1 = \{(1, (\{a, u_1\}, 1))\}$, which means that the system is at state 1 and there exists a control decision $\gamma_1 = \{a, u_1\}$ delayed in the control channel that will still be effective in one step. Then if we observe a, we have $\hat{x}_2 = \mathrm{NOR}_a(x_1) = \{(\delta(1, a), \mathrm{NX}((\gamma_1, 1))\} = \{(2, \{(\gamma_1, 0)\})\}$. From $\hat{x}_2$, if we make control decision $\gamma_2 = \{u_1\}$, then we have

- $(2, \{(\gamma_1, 0)\} \cup \{(\gamma_2, 1)\}) \in \mathrm{NUR}_{\gamma_2}(\hat{x}_2)$; and
- Since $u_1 \in \Gamma(\{(\gamma_1, 0), (\gamma_2, 1)\})$ and $\delta(2, u_1) = 3$, we have $(3, \mathrm{NX}(\{(\gamma_1, 0), (\gamma_2, 1)\}) \cup \{(\gamma_2, 1)\}) \in \mathrm{NUR}_{\gamma_2}(\hat{x}_2)$.

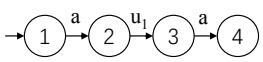

Fig. 1. System G with $\Sigma_o = \{a\}, \Sigma_c = \Sigma$ and $N_c = 1$

Therefore, we have $x_2 = \text{NUR}_{\gamma_2}(\hat{x}_2) = \{(2, \{(\gamma_1, 0), (\gamma_2, 1)\}), (3, \{(\gamma_2, 0), (\gamma_2, 1)\})\}$. ■

We are now ready to present our online state estimation algorithm. The main idea is to employ $\text{NUR}(\cdot)$ and $\text{NOR}(\cdot)$ 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 by: for any $\alpha \in \Sigma_o^*$ and $\sigma \in \Sigma_o$, we have

$$\hat{\mathcal{E}}_S(\epsilon) = \{(q_0, \emptyset)\} \tag{4}$$

$$\mathcal{E}_S(\alpha) = \text{NUR}_{S(\alpha)}(\hat{\mathcal{E}}_S(\alpha)) \tag{5}$$

$$\hat{\mathcal{E}}_S(\alpha\sigma) = \text{NOR}_{\sigma}(\mathcal{E}_S(\alpha)) \tag{6}$$

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

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

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

$$\theta(s) = \{(\gamma, N_c - n) : 0 \leq n \leq \min\{N_c, |s|\}, \gamma = S(P(s_{-n}))\} \tag{7}$$

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

$$\Gamma(\theta(s)) = S(P(s)) \cup S(P(s_{-1})) \cup \cdots \cup S(P(s_{-N_c})). \tag{8}$$

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

$$\mathcal{E}_S(\alpha) = \{(\delta(s), \theta(s)) \in \tilde{Q} : \exists s \in \mathcal{L}(S/G) \text{ s.t. } P(s) = \alpha\}. \tag{9}$$

For any set of extended states $x = \{(q_1, \theta_1), \cdots, (q_n, \theta_n)\}$, we denote by $Q(x)$ the set of all states in its first component, i.e., $Q(x) = \{q_1, q_2, \cdots, q_n\}$. 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 $Q(\mathcal{E}_S(\alpha)) = E_S(\alpha)$.

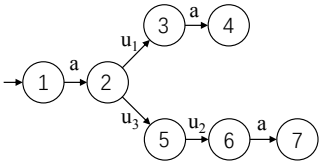

Fig. 2. System G with $\Sigma_c = \Sigma, \Sigma_o = \{a\}$ and $N_c = 1$.

Example 2: Let us consider the system G shown in Figure 2 with $\Sigma_c = \Sigma, \Sigma_o = \{a\}$ and $N_c = 1$. Suppose that the system is controlled by supervisor S defined by: $S(\epsilon) = \gamma_0$ and $S(a) = \gamma_1$, where $\gamma_0 = \{a, u_1, u_2, u_3\}$ and $\gamma_1 = \{a, u_1, u_3\}$. We now computed 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 $\mathcal{E}_S(a) = \text{NUR}_{\gamma_1}(\hat{\mathcal{E}}_S(a)) = \{(2, \{(\gamma_0, 0), (\gamma_1, 1)\}), (3, \{(\gamma_1, 0), (\gamma_1, 1)\}), (5, \{(\gamma_1, 0), (\gamma_1, 1)\})\}$. Note that state 6 is not contained in $\mathcal{E}_S(a)$ since $u_2 \notin \Gamma(\{(\gamma_1, 0), (\gamma_1, 1)\}) = \{a, u_1, u_3\}$.

On the other hand, by definition of $E_S(\cdot)$ and $\mathcal{L}(S/G)$, we have $\mathcal{L}(S/G) = \{\epsilon, a, au_1, au_1a, au_3\}$. Therefore, we have $E_S(a) = \{\delta(s) : s \in \mathcal{L}(S/G) s.t. P(s) = a\} = \{2, 3, 5\}$. This is consistent with our result that $E_S(a) = Q(\mathcal{E}_S(a))$. ■

IV. State Prediction and Choice of Control Decision

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

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

$$\Gamma_{\geq m}(\theta) = \bigcup_{i \in \{1, \dots, k\} : n_i \geq m} \gamma_i \tag{10}$$

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

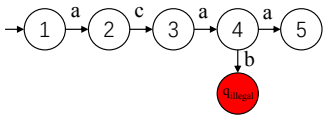

Fig. 3. System G with $\Sigma_c = \Sigma = \Sigma_o$ and $N_c = 1$.

the current channel configuration is θ, then we cannot prevent any string in $L_{uc}(\theta)$ from happening due to existing delayed control decisions.

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

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

The uncontrollable state prediction 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 uncontrollable state prediction 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 consider the system G shown in Figure 3 with $\Sigma_c = \Sigma = \Sigma_o$ and $N_c = 1$. Suppose we are now at extended state $\tilde{q} = (3, \theta)$, where $\theta = \{(\{a, c\}, 0), (\{b, c\}, 1)\}$, and we want to compute its uncontrollable state prediction $\text{USP}(\tilde{q})$. First, we have

$$\begin{aligned} L_{uc}(\theta) =& \overline{\Gamma_{\geq 0}(\theta)\Gamma_{\geq 1}(\theta)} = \overline{\{a, b, c\}\{b, c\}} \\ =& \overline{\{ab, bb, cb, ac, bc, cc\}} \end{aligned}$$

Therefore, we have

$$\text{USP}(\tilde{q}) = \{\delta(3, s) \in Q \colon s \in L_{uc}(\theta)\} = \{3, 4, 5, q_{illgeal}\}.$$

Therefore, we know that, we will unavoidably reach state $q_{illegal}$ starting from such extended state $\tilde{q} = (3, \theta)$. ■

Suppose that string $\alpha \in \Sigma_o^*$ is observed and the extended state estimate of the system (before the lastest 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

$$\text{USP}(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))) \subseteq Q_{good}. \tag{12}$$

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

V. Online Control Algorithm

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 Equation (12). 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 NUR 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 NOR operator. Again, we go to line 5 to choose a safe control decision and repeat the above procedure indefinitely.

Algorithm 1: Online-Control (G, N_c, Q_{good})

1 $\alpha \leftarrow \epsilon$, $\hat{\mathcal{E}}_S(\alpha) \leftarrow \{(q_0, \emptyset)\}$;
2 Go to line 5;
while *a new event* $\sigma \in \Sigma_o$ *is observed* **do**
3 $\quad \hat{\mathcal{E}}_S(\alpha\sigma) \leftarrow \text{NOR}_\sigma(\mathcal{E}_S(\alpha))$;
4 $\quad \alpha \leftarrow \alpha\sigma$;
5 $\quad$ Find a maximal control decision $\gamma \in \Gamma$ such that $\text{USP}(\text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))) \subseteq Q_{good}$;
6 $\quad$ Make control decision $S(\alpha) \leftarrow \gamma$;
7 $\quad \mathcal{E}_S(\alpha) \leftarrow \text{NUR}_\gamma(\hat{\mathcal{E}}_S(\alpha))$;

Remark 3: The maximal safety control decision in line 5 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., [22].

The following result shows the correctness of the proposed online supervisor.

Theorem 2: Let G be a DES and we denote by S_{online} the online supervisor defined in Algorithm 1. Then we have $\mathcal{L}(S_{online}/G) \subseteq K$.

Finally, we illustrate the proposed online control algorithm by the following example.

Example 4: Let us consider system G shown in Figure 4 with $\Sigma = \Sigma_c, \Sigma_o = \{a, b\}$ and $N_c = 1$. The specification language K is generated by the sub-automaton obtained by removing illegal states from G. For this example, we have $Q_H = Q_{good}$ as all events are controllable. We apply the proposed online control algorithm to compute control decision at each instant.

Initially, the algorithm start 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 $\gamma_0 = \{a, b, u_1, u_2, u_3\}$ is such a maximal control decision. Then the state estimate is updated to

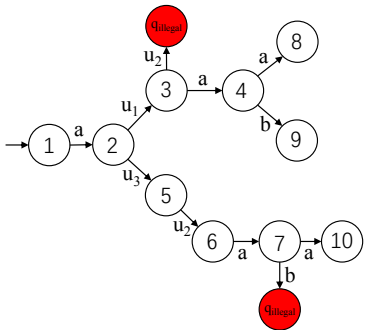

Fig. 4. System G with $\Sigma = \Sigma_c, \Sigma_o = \{a, b\}$ and $N_c = 1$.

$\mathcal{E}_S(\epsilon) = \text{NUR}_{\gamma_0}(\hat{\mathcal{E}}_S(\epsilon)) = \{(1, \{(\gamma_0, 1)\})\}$. 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 $q_{illegal}$ will be reached in the unobservable reach. Then the immediate state estimate upon next observable event a occurs is $\hat{\mathcal{E}}(aa) = \{(4, \{(\gamma_1, 0)\})\}$. By applying the online algorithm recursively again, we can obtain control decision $S(aa) = \gamma_2 = \{a, b, u_1, u_2, u_3\}$. ■

Remark 4: In [11], a supervisor called *predictive supervisor*, denoted by S_{pnc}, was proposed in order to exactly achieve K. Moreover, the authors also show that, when K is not exactly achievable, the synthesized supervisor is still safe. However, both state estimation and state prediction procedures of S_{pnc} are based on the specification K. Therefore, S_{pnc} is conservative as some infeasible behaviors are also considered. For example, in Example 4, the closed-loop language generated by our online supervisor is $\mathcal{L}(S_{online}/G) = \overline{\{au_3, au_1aa, au_1ab\}}$, while the predictive supervisor as defined in [11] achieves $\mathcal{L}(S_{pnc}/G) = \overline{\{au_3, au_1aa\}}$. Clearly, we see that S_{online} is strictly more permissive than S_{pnc}. Intuitively, this is because that the predictive supervisor estimates the state of the system based on K. Therefore, upon the occurrence of aa, S_{pnc} still needs to disable event b to avoid transition $7 \xrightarrow{b} q_{illgeal}$. However, by precisely estimating the state of the system using previous control information, our online supervisor knows that state 7 is not reachable since event u_2 was disabled in the previous step.

VI. Conclusion

In this paper, we proposed an online approach for solving the safety supervisory control problem of networked DES with control delays. To this end, we proposed the concept of extended states that augments the plant states with channel configurations. We presented techniques for online state estimation and prediction based on the extended state estimate. We showed that the proposed online supervisor is safe. We also illustrated by example that the proposed supervisor may achieve more permissive behavior than the predictive supervisor proposed in [11].

References

[1] J. Lunze, *Control Theory of Digitally Networked Dynamic Systems*. Springer, 2014.

[2] C. Nunes, M. Moreira, M. Alves, L. Carvalho, and J. Basilio, "Codiagnosability of networked discrete event systems subject to communication delays and intermittent loss of observation," *Discrete Event Dynamic Systems*, pp. 1–32, 2018.

[3] Y. Sasi and F. Lin, "Detectability of networked discrete event systems," *Discrete Event Dynamic Systems*, vol. 28, no. 3, pp. 449–470, 2018.

[4] M. Zgorzelski and J. Lunze, "A method for the synchronisation of networked discrete-event systems," in *13th International Workshop on Discrete Event Systems (WODES)*, pp. 444–451, 2016.

[5] X. Yin and S. Li, "Verification of opacity in networked supervisory control systems with insecure control channels," in *IEEE Conference on Decision and Control*, pp. 4851–4856, 2018.

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

[7] S.-J. Park and K.-H. Cho, "Delay-robust supervisory control of discrete-event systems with bounded communication delays," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 911–915, 2006.

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

[9] S. Shu and F. Lin, "Supervisor synthesis for networked discrete event systems with communication delays," *IEEE Transactions on Automatic Control*, vol. 60, no. 8, pp. 2183–2188, 2015.

[10] J. Komenda and F. Lin, "Modular supervisory control of networked discrete-event systems," in *13th International Workshop on Discrete Event Systems*, pp. 85–90, 2016.

[11] S. Shu and F. Lin, "Predictive networked control of discrete event systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4698–4705, 2017.

[12] B. Zhao, F. Lin, C. Wang, X. Zhang, M. Polis, and L. Wang, "Supervisory control of networked timed discrete event systems and its applications to power distribution networks," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 2, pp. 146–158, 2017.

[13] A. Rashidinejad, M. Reniers, and L. Feng, "Supervisory control of timed discrete-event systems subject to communication delays and non-fifo observations," in *14th International Workshop on Discrete Event Systems*, pp. 456–463, 2018.

[14] S. Takai and T. Ushio, "Effective computation of an $L_m(G)$-closed, controllable, and observable sublanguage arising in supervisory control," *Systems & Control Letters*, vol. 49, no. 3, pp. 191–200, 2003.

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

[16] X. Yin and S. Lafortune, "Synthesis of maximally permissive supervisors for partially observed discrete event systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 5, pp. 1239–1254, 2016.

[17] X. Yin and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 8, pp. 2140–2154, 2016.

[18] X. Yin and S. Lafortune, "Synthesis of maximally-permissive supervisors for the range control problem," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3914–3929, 2017.

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

[20] T. Ushio and S. Takai, "Nonblocking supervisory control of discrete event systems modeled by Mealy automata with nondeterministic output functions," *IEEE Trans. Autom. Contr.*, vol. 61, no. 3, pp. 799–804, 2016.

[21] X. Yin, "Supervisor synthesis for Mealy automata with output functions: A model transformation approach," *IEEE Transactions on Automatic Control*, vol. 62, no. 5, pp. 2576–2581, 2017.

[22] 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.

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

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