Coordination and Control of Distributed Discrete-event Systems subject to Sensor and Actuator Failures

Jin Dai, Student Member, IEEE, and Hai Lin, Senior Member, IEEE

Abstract

We study the coordination and control problem of distributed discrete-event systems with synchronous communication, in the presence of subsystems whose sensors and/or actuators may be affected by unexpected failures. We model sensor failures as permanent loss of observability of certain sensor events that belong to a subsystem, while characterize actuator failures as loss of controllability of the subsystems’ actuator events. The failure tolerance property requires that the distributed discrete-event systems satisfy a global specification prior to as well as after occurrences of potential failures. To prevent the failure-pruned subsystems from jeopardizing the fulfillment of the specification, we propose automaton-theoretic frameworks corresponding to the enforcement of sensor and actuator failure tolerance by incorporating learning-based supervisor synthesis and coordination approaches with appropriate post-failure control reconfiguration schemes. The effectiveness of the proposed frameworks is demonstrated by an illustrative example.

Index Terms

Discrete-event systems, distributed supervisory control, sensor and actuator failures, failure tolerant control, regular language learning

I. INTRODUCTION

Recent technological advances have greatly enhanced the ubiquitous application of large-scale engineering systems with spatially distributed physical interactions and interconnected communication topologies, such as power grids, transportation systems and communication networks. On the other hand, with increasingly sophisticated structure to achieve certain performance objections, complex engineering systems also become more vulnerable to faults and failures that may cause undesired outcomes. Therefore, how to guarantee safe and reliable operation of the large-scale engineering systems in the presence of potential system disturbances and failures is of major concern [1]. For such a pursuit, a variety of research efforts has been devoted to fault diagnosis and isolation (FDI) techniques and fault-tolerant control (FTC) strategies, see, e.g., [2], [3] and the references therein.

In this paper, we focus on the failure and fault tolerance of engineering systems whose behaviors are governed by operational rules that can be modeled by discrete-event systems (DESs) [4]. Since execution of various control commands on a system shows strong event-based features, DES models are widely used in the study of FDI and FTC.
problems of engineering systems. A great many methodologies have been developed to solve the FDI problems of DESs; and fruitful contributions have been made to centralized [5], decentralized [6–8], modular [9], [10], and distributed [11] system architectures. Nevertheless, how to appropriately handle the faults to avoid undesired consequences was not investigated in these established works.

Compared with extensive research on FDI of DESs, relatively less effort has been made to solve the FTC problem within the DES formalism. Lin [12] first considered the control problems of uncertain systems by modeling the uncontrolled system as a bank of candidate automata rather than a single nominal automaton, based on which robust and adaptive supervisors [13], [14] were designed. Rohloff [15] synthesized a robust supervisor to compensate for potential sensor failures; this work was further expanded by Sánchez and Montoya [16] where safety enforcement was accounted for in addition to failure tolerance. Other than sensor failures, Carvalho et al. [17] accounted for intrusion detection and control reconfiguration when actuator failures occurred. Wen et al. [18] modeled a fault as an uncontrollable event whose occurrence caused a faulty behavior, and provided a necessary and sufficient condition for the existence of a fault-tolerant supervisor. Paoli et al. [19] applied an “active” approach to tolerate potential event faults by introducing the notions of safe diagnosability and safe controllability. The guarantee of both fault-tolerance and safety was also fulfilled by Shu and Lin [20] via state-based feedback supervisors. The problem of reliable decentralized supervisory control was also considered in the literature [21], [22], where fault tolerance (termed as “reliability” therein) was defined with respect to failures of the supervisors rather than the systems. Iordache and Antsaklis [23] achieved resilience and fault-tolerance of Petri Nets (PN) by reconfiguring PN-based supervision with uncontrollable transitions. Robust control reconfiguration of PNs was also studied by Li et al. [24] with integer programming models. However, all these prior works were focused on a monolithic system, and failure tolerance of systems with more complex architectures against potential faults was not addressed.

We are therefore motivated to investigate the failure tolerance of complex systems in the DES framework. In particular, we concentrate our study on the coordination and control of distributed DESs with synchronous communication. Such a distributed DES is modeled as a parallel composition of multiple subsystems, each of which is characterized by a finite automaton over the local events. Our main objective is to investigate the performance of such distributed DESs in the presence of sensor and/or actuator failures that may occur in one or more subsystems. Roughly speaking, given a global control specification in the form of regular languages, a distributed DES can tolerate potential failures if by appropriate coordination and control policies, the specification can be fulfilled prior to as well as after occurrences of the failures. Formally, we model sensor failures as loss of observability of local sensor events; whereas actuator failures are described by loss of controllability of local actuator events. We expect to develop appropriate local supervisor reconfiguration schemes and coordination rules among the subsystems to guarantee the failure tolerance of the underlying DES. The main contributions of this paper can be summarized as follows:

1) Compared to the previous work [25] on fault-tolerant coordination of multi-agent systems in the DES framework, we consider not only cooperative task decomposition among the subsystems, but also the synthesis of local (nominal) supervisors. Furthermore, we modify the $L^*$ learning algorithm [26] to address the synthesis problem of the local supervisors even if explicit model of the subsystem is not known a priori.
2) We also investigate the failure tolerance of the system against potential sensor and/or actuator failures and aim to figure out how the subsystems reconfigure their local supervision policies such that the global specifications can still be maintained. To address this issue, we introduce an automaton-theoretic approach to systematically characterize each subsystem’s behavior in the presence of failures and propose control reconfiguration strategies corresponding to sensor and actuator failures, respectively.

3) Different from the various supervisory control approaches of distributed DESs \cite{27,29}, taking advantage of compositional verification techniques \cite{30}, we guarantee the coordination performance of the distributed DESs by exploiting an assume-guarantee paradigm \cite{30} with another modified \( L^* \) learning algorithm.

The remainder of this paper is organized as follows. In Section II we review the supervisory control theory of DESs. The effect of potential sensor and/or actuator failures of distributed DESs is described and modeled in Section III based on which we formulate the failure tolerant coordination and control of the underlying systems. In Section IV we propose a learning-based algorithm to synthesize nominal local supervisors for the distributed DESs. Section V develops an automaton representation of a subsystem influenced by sensor failures, and proposes a new supervisor synthesis framework to tolerate these sensor failures. In Section VI we solve the failure tolerant supervisor control problem with respect to actuator failures by introducing a control reconfiguration scheme. The proposed failure tolerant supervisory control strategies are combined in Section VII to address multiple failures for each subsystem. Section VIII justifies the coordinated performance of the distributed DESs by using a learning-based compositional verification algorithm. We apply the proposed failure tolerant coordination and control frameworks to a multi-robot system in order to examine their effectiveness respectively in Section IX. Finally, we end this paper with concluding remarks in Section X.

II. Preliminaries

The following notation and concepts are standard in the DES literature \cite{4}, \cite{31}. For a finite alphabet \( \Sigma \) of events, \( 2^\Sigma \) and \( |\Sigma| \) denote the power set and cardinality of \( \Sigma \), respectively. For two event sets \( \Sigma_1 \) and \( \Sigma_2 \), \( \Sigma_1 - \Sigma_2 \) denotes the set-theoretic difference of \( \Sigma_1 \) and \( \Sigma_2 \); \( \Sigma_1 \Delta \Sigma_2 \) denotes the symmetric difference of \( \Sigma_1 \) and \( \Sigma_2 \), i.e., \( \Sigma_1 \Delta \Sigma_2 = (\Sigma_1 - \Sigma_2) \cup (\Sigma_2 - \Sigma_1) \); and \( \Sigma_1 \Sigma_2 = \{ \sigma_1 \sigma_2 | \sigma_1 \in \Sigma_1, \sigma_2 \in \Sigma_2 \} \), where \( \sigma_1 \sigma_2 \) represents the concatenation of two elements \( \sigma_1 \) and \( \sigma_2 \).

A finite sequence \( w = \sigma_0 \sigma_1 \ldots \sigma_m \) composing of elements in \( \Sigma \) is called a word over \( \Sigma \). Let \( \Sigma^* \) denote the set of all finite-length words over \( \Sigma \), including the empty word \( \epsilon \). The length of a word \( w \in \Sigma^* \), written as \( |w| \), is the number of event symbols appeared in \( w \). A subset of \( \Sigma^* \) is called a language over \( \Sigma \). The prefix-closure of a language \( L \subseteq \Sigma^* \), denoted as \( \overline{L} \), is the set of all prefixes of words in \( L \), i.e., \( \overline{L} = \{ s \in \Sigma^* | (\exists t \in \Sigma^*) (st \in L) \} \). \( L \) is said to be prefix-closed if \( \overline{L} = L \). For two languages \( L_1, L_2 \subseteq \Sigma^* \), the right quotient is the collection of prefixes of words in \( L_1 \) with a suffix that belongs to \( L_2 \), i.e., \( L_1/L_2 = \{ s \in \Sigma^* | (\exists t \in L_2)(st \in L_1) \} \); whereas the left quotient of \( L_1 \) and \( L_2 \), defined as \( L_1 \setminus L_2 = \{ w \in \Sigma^* | (\exists s \in L_2)(sw \in L_1) \} \), is the set of words that occur in language \( L_1 \) after a word in \( L_2 \) has occurred.

An uncontrolled DES under consideration is modeled by a deterministic finite automaton (DFA) \( G = (Q, \Sigma, q_0, \delta, Q_m) \), where \( Q \) is the state set, \( \Sigma \) is the event set, \( q_0 \in Q \) is the initial state, \( \delta : Q \times \Sigma \rightarrow Q \) is the (partial) transition function.
and \( Q_m \subseteq Q \) is the set of marked (accepting) states. The transition function \( \delta \) can be extended to \( \delta : Q \times \Sigma^* \rightarrow Q \) in the usual manner \cite{4}. We use the notation \( \delta(q, s) \) to denote that the transition \( \delta(q, s) \) is defined. A DFA \( G \) is said to be trim if it is both accessible and coaccessible \cite{4}. The generated behavior of \( G \) is described by the prefix-closed language \( L(G) = \{ s \in \Sigma^* | \delta(q_0, s) \} \), and the accepted behavior of \( G \) is expressed by the marked language \( L_m(G) = \{ s \in \Sigma^* | s \in L(G), \delta(q_0, s) \in Q_m \} \). A language \( L \subseteq \Sigma^* \) is said to be regular if there exists a DFA \( G \) such that \( L_m(G) = L \). We will focus our study on regular languages.

Two DFAs \( G_1 \) and \( G_2 \) are coordinated by their parallel composition \( G_1 \parallel G_2 \) \cite{4}, \cite{30}, in which the two DFAs are “synchronized” on the common events. Since parallel composition is proved to be both commutative and associative \cite{4}, \cite{30}, coordination of more than two DFAs can be defined recursively.

Let \( \Sigma' \subseteq \Sigma \), the \textit{natural projection} \( P_{\Sigma'} : \Sigma^* \rightarrow \Sigma'^* \) from \( \Sigma^* \) to \( \Sigma'^* \) is defined such that: (i) \( P_{\Sigma'}(\epsilon) = \epsilon \); (ii) \( P_{\Sigma'}(\sigma) = \sigma \) if \( \sigma \in \Sigma' \); (iii) \( P_{\Sigma'}(\sigma) = \epsilon \) if \( \sigma \in \Sigma - \Sigma' \); and (iv) \( P_{\Sigma'}(s \sigma) = P_{\Sigma'}(s) P_{\Sigma'}(\sigma) \), \( \forall s \in \Sigma^* , \sigma \in \Sigma \). Given a language \( L \subseteq \Sigma^* \), \( P_{\Sigma'}(L) := \{ P_{\Sigma'}(s) \mid s \in L \} \). The corresponding inverse projection \( P_{\Sigma'}^{-1} : 2^{\Sigma'^*} \rightarrow 2^{\Sigma^*} \) of \( P_{\Sigma'} \) is defined as \( P_{\Sigma'}^{-1}(L) = \{ s \in \Sigma^* : P_{\Sigma'}(s) \in L \} \).

Let \( I = \{ 1, 2, \ldots, n \} \) be an index set. For \( n \) local event sets \( \Sigma_i(i \in I) \), we denote \( \Sigma = \bigcup_{i \in I} \Sigma_i \) as the \textit{global} event set, and let \( P_i(i \in I) \) be the natural projection from \( \Sigma^* \) to \( \Sigma_i^* \). The \textit{synchronous product} of a finite set of languages \( L_i \subseteq \Sigma_i^* (i \in I) \) is defined as \( \bigcap_{i \in I} P_i^{-1}(L_i) \). Clearly, the synchronous product operation of languages is commutative and associative. The following notion of separable languages plays an important role in the sequel of this paper.

\textbf{Definition 1 (Separable Languages):} \cite{32} For the local event sets \( \Sigma_i(i \in I) \), a language \( L \subseteq \Sigma^* \) is said to be \textit{separable} with respect to \( \{ \Sigma_i \}_{i \in I} \) if there exists a set of local languages \( L_i \subseteq \Sigma_i^* \), \( i \in I \) such that \( L = \bigcap_{i \in I} P_i(L_i) \).

It is shown in \cite{32} that \( L \subseteq \Sigma^* \) is separable with respect to \( \{ \Sigma_i \}_{i \in I} \) if and only if \( L = \bigcap_{i \in I} P_i^{-1}(L_i) \). It can also be shown that if a DES \( G = \bigcap_{i \in I} P_i(G_i) \), then \( L(G) = \bigcap_{i \in I} L(G_i) \).

\textbf{Remark 1:} Let \( D(\Sigma) = \{ (\sigma_1, \sigma_2) \in \Sigma \times \Sigma \mid \exists i \in I, \sigma_1, \sigma_2 \in \Sigma_i \} \) be the \textit{dependence relation} induced by \( \Sigma_i(i \in I) \). The \textit{independence relation} induced by \( \Sigma_i(i \in I) \) is then defined as \( I(\Sigma) = \Sigma \times \Sigma - D(\Sigma) \) \cite{33}. It follows from \cite{33} that a language \( L \subseteq L(G) \) admits a non-empty separable sublanguage if and only if \( I(\Sigma) \) is transitive.

We now recall the Ramadge-Wonham supervisory control theory of DESs \cite{13}, \cite{14}. For the system \( G \), we assume that \( \Sigma \) is partitioned into the set of controllable events \( \Sigma_c \) and the set of uncontrollable events \( \Sigma_{uc} \). The event set \( \Sigma \) can also be partitioned into the set of observable events \( \Sigma_o \) and the set of unobservable events \( \Sigma_{uo} \), i.e., \( \Sigma = \Sigma_c \cup \Sigma_{uc} = \Sigma_o \cup \Sigma_{uo} \). Let \( P_o \) be the natural projection from \( \Sigma^* \) to \( \Sigma_o^* \). A supervisor \cite{14} is a mapping \( S : P_o[L(G)] \rightarrow 2^{\Sigma} \) that observes the behavior of \( G \) and properly disables controllable events in \( \Sigma_c \). In practice, \( S \) is realized as another DFA over \( \Sigma \) whose states are all marked. \( S \) operates in parallel with \( G \) and the generated behavior of \( G \) under control of \( S \) is denoted as \( L(S \parallel G) \). To characterize the class of languages for which the supervisory control problem is solvable, we need the following properties.

\textbf{Definition 2:} \cite{4} A language \( L \subseteq \Sigma^* \) is said to be

- \textit{Controllable} with respect to \( G \) and \( \Sigma_{uc} \) if \( (\forall s \in \tilde{T})(\forall \sigma \in \Sigma_{uc})(ss \sigma \in L(G) \Rightarrow ss \sigma \in L) \); or equivalently, \( \tilde{T} \Sigma_{uc} \cap L(G) \subseteq \tilde{T} \).
Since we are interested in the generated behavior of the distributed DES in the presence of sensor and/or actuator failures. Finally, we present an outline of our approach for solving the problem.

III. PROBLEM FORMULATION

In this section, we first present the model of the distributed DESs subject to potential sensor and actuator failures that may occur in various subsystems. Afterwards, we formulate the failure tolerant coordination and control problem of the distributed DES in the presence of sensor and/or actuator failures. Finally, we present an outline of our approach for solving the problem.

A. Distributed DES with Sensor and Actuator Failures

We now consider a distributed DES $G$ that consists of $n$ subsystems (components), each of which is modeled by a DFA

$$G_i = (Q_i, \Sigma_i, q_{i,0}, \delta_i, Q_{i,m}) \quad (i \in I). \quad (1)$$

Since we are interested in the generated behavior of $G_i$, we do not need to define the marked states $Q_{i,m}$ explicitly.

For each subsystem $G_i(i \in I)$, let $\Sigma_i$ be partitioned into the set of locally controllable events $\Sigma_{i,c}$ and the set of locally uncontrollable events $\Sigma_{i,uc}$. Furthermore, $\Sigma_i$ is also partitioned into the set of locally observable events $\Sigma_{i,o}$ and the set of locally unobservable events $\Sigma_{i,uo}$. We associate with $\Sigma_{i,o}$ the natural projection $P_{i,o}: \Sigma_i \rightarrow \Sigma_{i,o}$.

In particular, it is assumed that $\Sigma_{i,o} = \Sigma_i$ when no sensor fails in $G_i(i \in I)$. The globally controllable and uncontrollable events are given by $\Sigma_c := \bigcup_{i \in I} \Sigma_{i,c}$ and $\Sigma_{uc} := \Sigma - \Sigma_c = \bigcap_{i \in I} \Sigma_{i,uc}$, respectively. It is further assumed that all the subsystems are coordinated through synchronous and ideal communication; therefore, the collective behavior of $G$ is captured by $G = \prod_{i \in I} G_i$. For a given event $\sigma \in \Sigma$, let $I_\sigma = \{i \in I|\sigma \in \Sigma_i\}$.

Potential failures of the subsystems are considered at this point. Here we focus our study on failures that may affect the functionalities of sensors and/or actuators. In this regard, we define $\Sigma_{i,s} \subset \Sigma_{i,uc}$ as the set of sensor events of subsystem $G_i(i \in I)$, whose observability is suspicious of failure. Since a practical engineering system is not able to alter its received sensor readings, it is reasonable to assume that all the sensor events are locally uncontrollable. Intuitively, by a sensor failure we mean that an event $\sigma_{i,s} \in \Sigma_{i,s}$ fails to maintain its observability.

**Definition 3 (Sensor Failures):** For each subsystem $G_i(i \in I)$, a sensor $\sigma_{i,s} \in \Sigma_{i,s}$ is said to be failed in $G_i$ if $\sigma_{i,s}$ becomes locally unobservable for $G_i$. A subset $\Sigma_{i,s}^f \subseteq \Sigma_{i,s}$ of local sensors is subject to failures if all the sensors $\sigma_{i,s} \in \Sigma_{i,s}^f$ fail.

On the other hand, actuators may refer to as a class of local events whose enablement status can be manipulated by the external supervisor. Therefore, we define $\Sigma_{i,a} \subset \Sigma_{i,c}$ as the set of actuator events of subsystem $G_i(i \in I)$, and define actuator failures as the loss of controllability of certain actuator events:
Definition 4 (Actuator Failures): For each subsystem \( G_i(i \in I) \), an actuator \( \sigma_{i,a} \in \Sigma_{i,a} \) is said to be failed in \( G_i \) if \( \sigma_{i,a} \) becomes locally uncontrollable for \( G_i \). A subset \( \Sigma_i^f \subseteq \Sigma_{i,a} \) of local actuators is subject to failures if all the actuators \( \sigma_{i,a} \in \Sigma_i^f \) fail.

In addition to the presence of possible failures, our proposed coordination and control strategy for the distributed DES also takes uncertainties of the subsystems into consideration. Specifically, an explicit DES model of \( G_i(i \in I) \) need not be given \textit{a priori}; instead, we apply a learning-based strategy to infer proper local supervisors for each subsystem.

B. Coordination and Control of Distributed DES subject to Failures

Our objective is to design a family of local nominal supervisors \( S_i \) as well as failure tolerant supervisors \( S_i^f \) corresponding to each subsystem \( G_i(i \in I) \) such that the both control performance and failure tolerance of the distributed system \( G \) can be achieved; that is, the coordination of all the supervised subsystems will “satisfy” a global specification, which is represented by a prefix-closed regular language over \( \Sigma \), prior to and after occurrences of possible sensor and/or actuator failures. We first formally define the satisfaction relation as follows.

Definition 5 (Property Satisfaction): Given a system modeled by a DFA \( M = (Q, \Sigma_M, q_0, \delta, Q_m) \) and a regular language (property) that is accepted by a DFA \( P = (Q_P, \Sigma_P, q_{0,P}, \delta_P, Q_{m,P}) \) with \( \Sigma_P \subseteq \Sigma_M \), the system \( M \) is said to satisfy \( P \), written as \( M \models P \), if and only if \( \forall t \in L_m(M) : P_P(t) \in L_m(P) \), where \( P_P \) denotes the natural projection from \( \Sigma_M \) to \( \Sigma_P \).

Remark 2: Note that when \( \Sigma_P = \Sigma_M \), the satisfaction relation of Definition 5 reduces to the language inclusion relation in the supervisory control theory. In case where \( Q_{m,P} = Q_P \), the marked languages considered in Definition 5 can be altered by generated languages.

Based on Definition 5, the failure tolerant coordination and control problem is formally stated as follows:

Problem 1: Given a distributed DES \( G \) that consists of \( n \) subsystems \( G_i(i \in I) \), along with the local sensor events \( \Sigma_{i,s} \) and actuator events \( \Sigma_{i,a} \), one needs to solve the following problems for a nonempty and prefix-closed global specification \( L \subseteq \Sigma^* \) whose independence relation (cf. Remark 1) is transitive:

1) Specification decomposition: systematically find a nonempty and locally feasible specification \( L_i \) for each subsystem \( G_i(i \in I) \);

2) Nominal local supervision: synthesize the local supervisors \( S_i(i \in I) \) such that \( L(S_i||G_i) \models L_i \) when no failures are detected;

3) Sensor failure tolerant control: when local sensors \( \Sigma_i^f \subseteq \Sigma_{i,s} \) fail in \( G_i(i \in I) \), synthesize a corresponding local supervisor \( S_i^f \) such that \( L(S_i^f||G_i^{F,s}) \models L_i \), where \( G_i^{F,s} \) denotes the post-failure subsystem \( G_i \).

4) Actuator failure tolerant control: when local actuators \( \Sigma_i^f \subseteq \Sigma_{i,a} \) fail in \( G_i(i \in I) \), synthesize a corresponding local supervisor \( S_i^a \) such that \( L(S_i^a||G_i^{F,a}) \models L_i \), where \( G_i^{F,a} \) denotes the subsystem \( G_i \) with actuator faults.

5) Distributed coordination: \( \big|\big|_{i \in I} L_i \models L \) can be held regardless of potential failures.
C. Overview of the Proposed Approach

Without loss of generality, we assume that $L \subseteq L(G)$. To pursue a distributed solution of Problem 1, we propose a three-layered formal synthesis framework that accounts for synthesis of both nominal supervisors and post-failure supervisory rules via “divide-and-conquer”. The synthesis procedure is accomplished by iteratively executing the framework shown in Fig. 1. Note that to address the potential uncertainty from system dynamics, the $L^*$ learning algorithm [26] is improved and the modified algorithm is utilized in the proposed framework.

1) Decomposition: Obtain a nonempty, prefix-closed and locally feasible specification $L_i$ for each subsystem $G_i (i \in I)$.

2) Nominal supervisor synthesis: given the local specification $L_i$, we modify the $L^*$ learning algorithm [26] to automatically synthesize a local supervisor $S_i$ for $G_i$ such that $L(S_i || G_i) = \sup C_i(L_i)$ when no sensor or actuator failures are detected, where $\sup C_i(L_i)$ stands for the supremal controllable sublanguage [4] with respect to $G_i$ and $\Sigma_{i,uc}$.

3) Post-failure reconfiguration: we assume that sensor and/or actuator failures can be detected immediately after their occurrences. Following the detection of the failure, we first construct post-failure DFA models corresponding to the subsystems whose evolution is affected by the failures. Next, appropriate failure tolerant supervisors $S_i^{F,s}$ (resp., $S_i^{F,a}$) are synthesized based on the post-failure model $G_i^{F,s}$ (resp., $G_i^{F,a}$) as well as the post-failure event sets $\Sigma_i^{F,s}$ (resp., $\Sigma_i^{F,a}$) for subsystem $G_i (i \in I)$. The joint effort of both nominal and post-failure supervisors prevents the behavior of the failed subsystems from violating the local specification.

4) Compositional verification: A compositional verification procedure [30] is deployed to justify whether or not all the supervised subsystems can jointly accomplish the global specification. For such a pursuit, we apply the assume-guarantee paradigm by setting the controlled subsystem $M_i (i \in I)$ to be the component modules and $L$ as the property to be verified. If the global specification fails to be satisfied, the verifier returns a counterexample $t \in \Sigma^*$ that indicates the violation of $L$. Such a counterexample $t$ is utilized for the re-synthesis of the local supervisors until no more counterexamples are detected.
IV. SYNTHESIS OF NOMINAL SUPERVISORS VIA LEARNING

This section concerns with the synthesis of nominal supervisors for each subsystem $G_i (i \in I)$ when no failures are detected. To cope with possible lack of prior knowledge of the DFA models $G_i (i \in I)$, we introduced a learning-based algorithm for the synthesis of nominal supervisors.

A. Initial Assignment of Local Specifications

The iterative execution of the proposed failure tolerant coordination and control framework shown in Fig. [1] initializes local specifications $L_i (i \in I)$ by taking natural projections of the global specification, i.e., $L_i = P_i (L)$. The local feasibility of the projected language $L_i$ is enforced by the following theorem.

**Theorem 1:** [34] Given the DES $G = \big||_{i \in I} G_i$, and a non-empty and prefix-closed specification $L \subseteq L(G)$, the language $L_i := P_i (L) (i \in I)$ satisfies that $L_i \subseteq L(G_i)$ and $L_i = \overline{L_i}$.

B. Essentials of the $L^*$ Learning Algorithm

We start the introduction of the learning-based supervisor synthesis algorithm with the basic ingredients of the $L^*$ learning algorithm (abbreviated as the $L^*$ algorithm in the sequel) developed by Angluin [26]. The algorithm infers an unknown regular language $U \subseteq \Sigma^*$ and constructs a minimal DFA that accepts it. The $L^*$ algorithm interacts with a Minimally Adequate Teacher, henceforth referred to as the Teacher, that answers two types of questions: the first type is referred to as membership queries, i.e., the Teacher justifies whether or not a word $s \in \Sigma^*$ belongs to $U$; the second type is a conjecture, in which the $L^*$ algorithm constructs a DFA $M$ and the Teacher determines whether or not $L_m (M) = U$.

The $L^*$ algorithm maintains information about a finite subset of words over $\Sigma$ and classifies them as either members or non-members of $U$. Such membership status is recorded by an observation table $(S, E, T)$, where $S$ and $E$ are a set of prefixes and suffixes over $\Sigma$, respectively; and $T : (S \cup S\Sigma) E \rightarrow \{0, 1\}$ is the membership function that maps words in $s \in (S \cup S\Sigma)$ onto 1 if they are members of $U$, otherwise it returns 0. The entry of the observation table in the row labeled by $s \in S \cup S\Sigma$ and column labeled by $e \in E$ is evaluated by $T(se)$. The row function $row(s)$ denotes the finite function $f$ from $E$ to $\{0, 1\}$ defined by $f(e) = T(se)$. The $L^*$ algorithm aims to construct a closed and consistent observation table during each iteration.

**Definition 6 (Closeness and Consistency):** [26] An observation table is said to be closed if $(\forall s \in S)(\forall \sigma \in \Sigma) [\exists s' \in S : row(\sigma s) = row(s')]$. It is said to be consistent if $(\forall s_1, s_2 \in S : row(s_1) = row(s_2))(\forall \sigma \in \Sigma)[row(s_1\sigma) = row(s_2\sigma)]$.

If an observation table is not closed, then $s\sigma$ is added to $S$ and $T$ is updated to make it closed where $s \in S$ and $\sigma \in \Sigma$ are the elements for which no $s' \in S$ exists. If an observation table is not consistent, then there exist $s_1, s_2 \in S$, $\sigma \in \Sigma$ and $e \in E$ such that $row(s_1) = row(s_2)$ but $T(s_1\sigma e) \neq T(s_2\sigma e)$; to make it consistent, $\sigma e$ is added to $E$ and $T$ is updated. Once an observation table is both closed and consistent, a candidate DFA $M(S, E, T) = (Q, \Sigma, q_0, \delta, Q_m)$ is constructed as follows: $Q = \{row(s) : s \in S\}$, $q_0 = row(\epsilon)$,

\[1\] By “minimal” we mean that the obtained DFA contains the least number of states.
\( Q_m = \{ \text{row}(s) : (s \in S) \land (T(s) = 1) \} \), and \( \delta(\text{row}(s), \sigma) = \text{row}(s\sigma) \). The Teacher takes \( M \) as a conjecture. If the conjecture is correct, i.e., \( L_m(M) = U \), the Teacher returns “True” with the current DFA \( M \); otherwise, the Teacher returns “False” with a counterexample \( c \in L_m(M) \Delta U \). The \( L^* \) algorithm adds all the words in \( [c] \) to \( S \) and iterates the entire procedure to update a new closed and consistent observation table.

The following theorem asserts that the sequence of DFAs constructed by the \( L^* \) algorithm is strictly increasing in the number of states.

**Theorem 2:** \(^{(26)}\) If \((S, E, T)\) is a closed and consistent observation table and \( M(S, E, T) \) is a DFA constructed from \((S, E, T)\). Let \((S', E', T')\) be the updated closed and consistent observation table if a counterexample \( t \) is added to \((S, E, T)\). If \( M(S, E, T) \) has \( n \) states, then the DFA \( M(S', E', T') \) constructed from \((S', E', T')\) has at least \( n + 1 \) states.

The correctness and finite convergence of the \( L^* \) algorithm is guaranteed by the following theorem.

**Theorem 3:** \(^{(26)}\) Given any Teacher presenting an unknown regular language \( U \subseteq \Sigma^* \), the \( L^* \) algorithm eventually terminates and outputs an DFA isomorphic to the minimal one accepting \( U \). Moreover, if \( n \) is the number of states of the minimal DFA accepting \( U \) and \( m \) is an upper bound on the length of any counterexample provided by the Teacher, then the total running time of the \( L^* \) algorithm is bounded by a polynomial in \( m \) and \( n \).

### C. The \( L^*_{LS} \) Synthesis Algorithm

Next, we proceed to the synthesis of the maximally permissive \(^{(4)}\) supervisor \( S_i \) for an assigned specification \( L_i \). As suggested by Definition \(^{(2)}\) applying directly the \( L^* \) algorithm to synthesize the nominal supervisor for subsystem \( G_i(i \in I) \) becomes infeasible since full knowledge of \( G_i \) may not be given a priori, rendering the controllability test of \( L_i \) to be impractical. To address this issue, we modify the \( L^* \) algorithm and propose a learning-based algorithm, namely the \( L^*_{LS} \) algorithm, to learn \( \sup C_i(L_i) \) for subsystem \( G_i \).

The nominal supervisor \( S_i \) is synthesized on the basis of illegal behaviors. A word \( s \in L(G_i) \) is said to be an illegal behavior if \( s \notin L_i \). A word \( st \in L(G_i) \) is said to be uncontrollably illegal if \( s \in L_i \), \( t \in \Sigma_{i,uc}^* \) and \( st \notin L_i \); that is, \( st \) becomes illegal due to an uncontrollable suffix \( t \). Let \( C_i \) be the collection of uncontrollably illegal behaviors of \( G_i(i \in I) \). Define \( D_{uc}() \) as an operator that collects words which are formed by discarding the uncontrollable suffixes of words in \( C_i \), i.e., \( D_{uc}(C_i) = \{ s \in L(G_i) | (\exists t \in \Sigma_{i,uc})[st \in C_i]\} \). Specifically, let \( C^j_i \) denote the set of uncontrollably illegal behavior after the \( j \)-th update of the observation table in the \( L^*_{LS} \) algorithm; thus \( C^{j+1}_i = C^j_i \cup \{ s_j \} \) if a new uncontrollably illegal behavior \( s_j \) is added to \( C^j_i \).

Different from the \( L^* \) algorithm, the \( L^*_{LS} \) algorithm relies on a family of dynamical and conditional membership queries. Let \( T^1_i(j \in \mathbb{N}) \) denote the membership function for \( G_i(i \in I) \). Initially, the Teacher justifies the membership of a word \( t \in \Sigma_i^* \) with respect to \( L_i \),

\[
T^1_i(t) = \begin{cases} 
0, & \text{if } t \notin L_i, \\
1, & \text{otherwise},
\end{cases}
\]  

\(^{(2)}\)The subscript “LS” stands for “local synthesis”.
and for \( j \geq 2 \), the Teacher provides answers for the following conditional queries:

\[
T_j^i(t) = \begin{cases} 
0, & \text{if } T_j^{i-1}(t) = 0 \text{ or } t \in D_{u_i}(C_j^i)\Sigma_i^*, \\
1, & \text{otherwise.} 
\end{cases}
\]  

Furthermore, the lack of prior knowledge of \( G_i \) inspires us to introduce the following sequence \( \{K_j\}(j \in \mathbb{N}) \) to facilitate the generation of counterexamples.

\[ K_1 := L_i, \]
\[ K_j := K_{j-1} - D_{u_i}(C_j^i)\Sigma_i^*. \]  

With slightly abusing the notations, we also use \( K_j \) to denote the DFA that recognizes the language \( K_j(j \in \mathbb{N}) \). Let \( M_j = M(S^j, E^j, T_j^i) \) be the DFA that is consistent with membership function \( T_j^i \), a word \( t \in \Sigma_i \) shall be presented as a counterexample by the \( L_{LS}^* \) algorithm if

\[ t \in L(M_j) \Delta K_j. \]  

Details of the \( L_{LS}^* \) algorithm are presented in Algorithm 1, which can be viewed as an execution of the \( L^* \) algorithm with membership queries (2)-(3) and counterexamples generated from (5).

The correctness and finite convergence of the \( L_{LS}^* \) algorithm are formally summarized in the following theorem.

**Theorem 4:** [34] For a nonempty and prefix-closed local specification \( L_i \subseteq L(G_i) \) for \( G_i(i \in I) \), the \( L_{LS}^* \) algorithm with dynamical membership queries (2)-(3) and counterexample generated from (5) converges to a local supervisor \( S_i \) such that \( L(S_i||G_i) = \sup C_i(L_i) \). Moreover, the \( L_{LS}^* \) algorithm can always construct the correct \( S_i \) within a finite number of iterations.

**Remark 3:** It follows from Theorem 3 that the \( L_{LS}^* \) algorithm also admits a polynomial-time computational complexity. Different from the conventional supervisor synthesis algorithms presented in [31], the \( L_{LS}^* \) algorithm requires no prior knowledge of the uncontrolled system.

V. SENSOR FAILURE TOLERANT SUPERVISORY CONTROL

Behaviors of each subsystem \( G_i \) \((i \in I)\) under the nominal control policy \( S_i \) may deviate from the assigned specification \( L_i \) when \( G_i \) encounters with possible sensor and/or actuator failures. In this section, we deal with the synthesis problem of failure tolerant local supervisors for the subsystems whose behaviors are influenced by sensor failures. As shown in Fig. 1, our proposed local supervisory control scheme to tolerate sensor failures consists of two major ingredients: construction of a DFA model \( G_i^{F,s} \) of \( S_i||G_i(i \in I) \) when a sensor fails; and the synthesis of a failure tolerant supervisor \( S_i^* \) when a sensor failure occurs.

A. Behaviors of DESs subject to Sensor Failures

We first characterize the behaviors of a supervised subsystem under the influence of sensor failures. Towards this end, we use

\[ G_i^0 := S_i||G_i \]
Algorithm 1: $L_{LS}^*$ Algorithm

Input: $L_i$, $\Sigma_{i,c}$ and $\Sigma_{i,uc}$

Output: $S_i : L(S_i) = L(S_i) = \sup C_i(L_i)$

1: $S \leftarrow \{\epsilon\}$, $E \leftarrow \{\epsilon\}$, $j \leftarrow 1$
2: Construct $T^j_i(S,E,T)$ with membership queries (2)
3: repeat
4: while $T^j_i(S,E,T)$ is not closed or consistent do
5: if $T^j_i$ is not closed then
6: find $s \in S$, $\sigma \in \Sigma_i$ such that $\forall s' \in S : \text{row}(s'\sigma) \neq \text{row}(s)$
7: $S \leftarrow S \cup \{s\sigma\}$
8: extend $T^j_i$ to $(S \cup \Sigma_i)E$ using membership queries (2) or (3)
9: end if
10: if $T^j_i$ is not consistent then
11: find $s_1,s_2 \in S$, $\sigma \in \Sigma_i$ and $e \in E$ such that $\text{row}(s_1) = \text{row}(s_2)$ but $T(s_1\sigma e) \neq T(s_2\sigma e)$
12: $E \leftarrow E \cup \{\sigma e\}$
13: extend $T^j_i$ to $(S \cup \Sigma_i)E$ using membership queries (2) or (3)
14: end if
15: end while
16: $M_j \leftarrow M(S^j_i,E^j_i,T^j_i)$
17: if the Teacher presents a counterexample $t \in \Sigma^*_i$ according to (5) then
18: $S \leftarrow S \cup T$
19: $j \leftarrow j + 1$
20: extend $T^j_i$ to $(S \cup \Sigma_i)E$ using membership queries (2) or (3)
21: end if
22: until the Teacher generates no more counterexamples.
23: return $S_i = M_j$

To denote the closed-loop subsystem $G_i(i \in I)$ under the supervision of the local supervisor $S_i$, and let $L_i^0 = L(G_i^0) = \sup C_i(L_i)$ be the language generated by $G_i^0$. Furthermore, we assume that $G_i^0$ can be represented by the following DFA over $\Sigma_i$:

$$G_i^0 = (Q_i^0, \Sigma_i, q_{i,0}, \delta_i^0).$$

Note that the marked states are omitted here since all the states in $Q_i^0$ should be marked. Without loss of generality, we assume that $\Sigma_{i,s}$ includes $N_i$ suspicious sensors, i.e.,

$$\Sigma_{i,s} = \{\sigma_{i,1}, \sigma_{i,2}, \ldots, \sigma_{i,N_i}\}.$$
The presence of $\Sigma_{i,s}$ in fact induces $N_i$ potential failure modes of $G_i^0$. The $k$-th failure mode corresponding to a failed sensor $\sigma_{i,k}$ is captured by the following DFA

$$G_i^k = (Q_i^k, \Sigma_i, \delta_i^k), \quad k = 1, 2, \ldots, N_i,$$

where $Q_i^k$ can be viewed as a copy of $Q_i^0$ such that for each $q_i^0 \in Q_i^0$ there exists a corresponding $q_i^k \in Q_i^k$, and $\delta_i^k$ is formally defined as $\delta_i^k(q_i^k, \sigma) = q_i^k$ if and only if $\delta_i^0(q_i^0, \sigma) = q_i^0$. It is worth pointing out that we do not specify the initial state $q_i^k$ of the DFA $G_i^k$ since it depends on at which state of $G_i^0$ the sensor failure $\sigma_{i,k}$ occurs.

We assume that a sensor can only fail in the nominal mode. To govern the transitions from the nominal mode to one of the failure modes, we introduce an extra set of sensor failure flag events

$$\Sigma_{i,s}^f = \{f_{i,1}, f_{i,2}, \ldots, f_{i,N_i}\}$$

such that $f_{i,k}$ drives the controlled system component from the nominal mode $G_i^0$ to $G_i^k$. To pursue succinct notations, we also use $f_{i,k}$ to denote an associated mapping

$$f_{i,k} : Q_i^0 \rightarrow Q_i^k, \quad k = 1, 2, \ldots, N_i$$

which indicates the circumstance that the nominal subsystem $G_i^0(i \in I)$ encounters with a failed sensor $\sigma_{i,k}$. In particular, $f_{i,k}(q_{i,l}^0) = q_{i,l}^k$ implies that the sensor failure $\sigma_{i,k}$ occurs when $G_i^0$ is at state $q_{i,l}^0$ and the initial state of the $k$-th failure mode $G_i^k$ should be its copy, i.e., $q_{i,l}^k$. Thus, $G_i(i \in I)$ under the local control rule $S_i$ in the presence of sensor failures in $\Sigma_{i,s}$ turns to be a combination from the following bank of DFAs:

$$G_i = \{G_i^0, G_i^1, \ldots, G_i^{N_i}\}.$$

Although it is convenient to use elements in $G_i$ to describe the behavior of $G_i(i \in I)$ prior to as well as after the occurrence of a sensor failure, we can also view the supervised behavior of $G_i(i \in I)$ as part of the language generated by the following unified trim DFA

$$\overline{G}_i = (\overline{Q}_i, \Sigma_i, q_i^0, \overline{\delta}_i),$$

where $\overline{Q}_i = Q_i^0 \cup (\cup_{k=1}^{N_i} Q_i^k)$, $\Sigma_i = \Sigma_i \cup \Sigma_{i,s}^f$, and $\overline{\delta}_i = (\cup_{k=0}^{N_i} \delta_i^k) \cup \{(q_i^0, f_{i,k}, q_{i,l}^k) : f_{i,k}(q_{i,l}^0) = q_{i,l}^k\}$.

**B. Synthesis of Sensor Failure Tolerant Supervisors**

Based on the aforementioned characterization of the controlled behavior of each subsystem $G_i(i \in I)$ before and after the occurrence of a single sensor failure, we now propose component-wise failure tolerant control schemes for the distributed DES $G$ when some of the subsystems are affected by local sensor failures.

Motivated by the assumption that $\Sigma_{i,s} \subseteq \Sigma_{i,uc}$ (cf. Section III.B), we apply a conditional “modular” supervisory control approach for the design of failure tolerant supervisor $S_i^f$, which is illustrated in Fig. 2. The failure tolerant control schemes can be stated as follows:

1) When no sensor in $\Sigma_{i,s}$ fails, $\Sigma_{i,o} = \Sigma_i$, we only activate the nominal supervisor $S_i$ for $G_i(i \in I)$ to assure the fulfillment of the local specification.
2) When a sensor failure $\sigma_{i,k}$ is detected, $\{\sigma_{i,k}\}$ becomes unobservable, the controlled subsystem evolves from the nominal mode $G_i^0$ to the $k$-th failure mode $G_i^k (k = 1, 2, \ldots, N_i)$. In this case, we activate the corresponding failure tolerant supervisor $S^{\text{fs}}_i$, which collaborates with $S_i$ simultaneously to control $G_i$. Note that $S^{\text{fs}}_i$ makes control decision through the natural projection $P_{i,o}$ induced by $\sigma_{i,k}$.

For simplicity of presentation, we assume that a sensor $\sigma_{i,k}$ fails after a word $t \in L_i^0$ has been generated by the nominally supervised subsystem $G_i^0$ and that the corresponding state in $G_i^0$ at which the sensor fails is $q_{i,0}^0$. We also assume that $f_{i,k}(q_{i,0}^0) = \overline{q}_{i,0}$. Therefore, the DFA representation of $G_i$’s controlled behavior before and after the occurrence of $\sigma_{i,k}$ can be extracted from $G_i$ in the following manner:

$$G_i^{F,s} = (Q_i^0 \cup Q_i^k, \Sigma_i \cup \{f_{i,k}\}, q_{i,0}^0, \delta_{i,0}^0 \cup \delta_{i,k}^0 \cup \{(q_{i,0}^0, f_{i,k}, q_{i,0}^k)\}),$$

where the event set $\Sigma_i^{F,s} = \Sigma_i \cup \{f_{i,k}\}$ of the subsystem $G_i^{F,s} (i \in I)$ with the sensor failure satisfies the following partitions:

$$\Sigma_i^{F,s} = \Sigma_i^{c,s} \cup \Sigma_i^{uc,s} \cup \{f_{i,k}\},$$

and

$$\Sigma_i^{F,s} = (\Sigma_i - \{\sigma_{i,k}\}) \cup \{f_{i,k}\}; \Sigma_i^{c,s} = \{\sigma_{i,k}\}.$$  \hspace{1cm} (6)

We use an example to illustrate the construction procedure of the post-failure model of a supervised subsystem.

**Example 1:** The DFA model of a closed-loop subsystem $G_i^0$ under consideration is depicted in Fig. 3. We assume that the sensor event set is given by $\Sigma_i = \{a\}$ and all the events are locally observable when no sensor fails.

![Fig. 3: The nominal subsystem $G_i^0$.](image)

We then study the case where the sensor $a$ fails when $G_i^0$ evolves at state $q_{i,1}^0$, which is denoted as the failure mode $G_i^1$. In this case, we use $f_{i,a}$ to represent the sensor failure flag event associated with $a$. According to the
For the non-empty and prefix-closed specification \( L_i^0 \) with\( i \in I \) and a word \( t \in L_i^0, L_i^0 = L_i^0 \setminus \{ t \} \), is also prefix-closed.

\begin{proof}
It suffices to prove that \( \overline{T_i} \subseteq L_i^0 \). By definition of the left quotient of languages, \( L_i^0 = L_i^0 \setminus \{ t \} = \{ u \in \Sigma^* | tu \in L_i^0 \} \). Thus, for any \( u' \in \pi, tu' \in \overline{T_i} \subseteq L_i^0 \) due to the prefix-closeness of \( L_i^0 \), which implies that \( u' \in L_i^0 \setminus \{ t \} \). Therefore, \( \overline{T_i} \subseteq L_i^0 \) and the proof is completed.
\end{proof}

Given the DFA representation \( G_i^k \) of the failure mode of \( G_i^k(i \in I) \), we require the partial-observation supervisor \( S_i^* \) to fulfill the local control task \( L_i^* \), i.e., \( L(S_i^* || G_i^k) = L_i^* \). Such a synthesis objective is accomplished via two steps: first, we synthesize a complete-observation supervisor \( S_i' \) by using the \( L_{LS} \) algorithm to achieve the supremal controllable sublanguage of \( L_i^* \) with respect to the locally uncontrollable events \( \Sigma_{i,uc} \), denoted as \( \text{sup} \ C_i^*(L_i^*) \). Next, we incorporate \( S_i' \) with the locally observable events \( \Sigma_{i,o} \) to synthesize the desired \( S_i^* \).

The generation of \( S_i' \) is similar to \( S_i \) and we omit it here. Without loss of generality, we assume that \( S_i' \) can be represented by the following DFA

\[ S_i' = (Q_i', \Sigma_i \cup \{ f_{i,k} \}, q_i^0, \delta_i' ) \]

The failure tolerant supervisor \( S_i^*(i \in I) \) is then constructed on the basis of \( S_i' \). For each \( q' \in Q_i' \), we define

\[ \text{Act}(q') = \{ \sigma \in \Sigma_i \cup \{ f_{i,k} \} | \delta_i'(q', \sigma) \} \]

as the set of active events in \( S_i' \) at state \( q' \). For any subset \( Q' \subseteq Q_i' \), the set of \( \sigma \)-reachable states from \( Q' \) with an observable event

\[ \text{OR}(Q', \sigma') = \{ q \in Q_i' | (\exists q' \in Q_i') [ q' = \delta_i'(q', \sigma') ] \} \]

and the unobservable reach of states in \( Q' \) is defined as

\[ \text{UR}(Q') = \{ q \in Q_i' | (\exists q' \in Q_i') (\exists w \in \Sigma_{i,uo} | q = \delta_i'(q', s)) \} \].
With the aforementioned notations, the fault-tolerant supervisor $S_i^s$ is then given by a trim DFA
\[
S_i^s = \text{trim}(Q_i^s, \Sigma_i^{F,s}, q_{i,0}^s, \delta_i^s),
\]  
with \text{trim} being the operator to extract the trim part of a DFA. The ingredients of $S_i^s$ are listed as follows:
\[
Q_i^s = 2Q_i', q_{i,0}^s = UR(\{q_{i,0}'\}) \text{ and }
\forall q_i^s \in Q_i', \sigma \in \Sigma_i^{F,s}: \delta_i^s(q_i^s, \sigma) = UR(OR(q_i^s, \sigma)).
\]

Remark 4: The construction of $S_i^s$ from $S_i'$ follows a similar synthesis pattern of the online supervisory control approach proposed in [35], which admits a computational complexity of $O(|Q_i^s||Q_i'|)$. It is also shown in [35] that $L(S_i^s||G_i^k)$ is a controllable and observable sublanguage of $L_i'$, indicating that $S_i^s$ will not drive the post-failure subsystem to violate the local specification.

We slightly abuse the notations and denote $L(S_i^s||G_i^{F,s}(i \in I))$ as the closed-loop behavior of $G_i(i \in I)$ under the influence of the failed sensors in $\Sigma_i,s$. The following theorem claims that, for each $G_i(i \in I)$ with a possible sensor failure from $\Sigma_i,s$, the joint effort of local nominal supervisor $S_i$ and failure-tolerant supervisor $S_i^s$ will not violate the local specification $L_i^0$.

**Theorem 5:** For the subsystem $G_i(i \in I)$ with supervisors $S_i$ and $S_i^s$, $L(S_i^s||G_i^{F,s}(i \in I)) = L_i^0$.

**Proof:** One can infer from Fig. 2 that the closed-loop behavior of $S_i^s||G_i^{F,s}(i \in I)$ shall consist of three parts: i) the nominal behavior prior to a sensor failure; ii) the generation of the corresponding sensor failure flag event; iii) the post-failure controlled behavior $S_i^s||G_i^{F,s}$ after the activation of $S_i^s$. We assume without loss of generality that a sensor fault $\sigma_{i,k}$ occurs after $G_i(i \in I)$ has executed a behavior $t \in L_i^0$. Before the sensor $\sigma_{i,k}$ becomes faulty, we have
\[
\mathcal{F} \subseteq L(S_i^s||G_i^{F,s}) = L(S_i||G_i) = L_i^0.
\]

After the sensor failure flag event $f_{i,k}$ is released, $G_i^0$ enters the $k$-th $G_i^k$; thus any behavior $t'$ generated by the underlying subsystem afterwards satisfies that
\[
\mathcal{F} \subseteq L(S_i^s||G_i^{F,s}) \subseteq L_i' = L_i^0 \setminus \{t\},
\]
which implies that the overall behavior performed by $S_i^s||G_i^{F,s}(i \in I)$ is given by the following collection:
\[
L(S_i^s||G_i^{F,s}) = \{\mathcal{T}_{i,k}t' \mid t \in L_i^0, t' \in L_i^0 \setminus \{t\}\}.
\]

Since $f_{i,k} \notin \Sigma_i$ for all $i \in I$, it immediately follows from Definition 5 that
\[
P_{f,i}[L(S_i^s||G_i^{F,s})] = \{\mathcal{T}_t \mid t \in L_i^0, t' \in L_i^0 \setminus \{t\}\} \subseteq L_i^0 = L_i^0,
\]
with $P_{f,i}$ being the natural projection from $(\Sigma_i \cup \{f_{i,k}\})^*$ to $\Sigma_i^*$, which is equivalent to $L(S_i^s||G_i^{F,s}) = L_i^0$. The proof is completed. □
VI. Active Supervisor Reconfiguration for Actuator Failures

In addition to sensor failures, the closed-loop performance of the subsystems that belong to $G$ may deviate from the assigned specifications due to the presence of actuator failures. In this section, we aim to develop local actuator failure tolerant control schemes. Specifically, we employ a supervisor reconfiguration and switching mechanism to actively cope with actuator failures.

A. DES Characterization of Actuator Faults

From Definition 4, an actuator failure results in the loss of controllability of a local actuator event. Different from sensor faults, unexpected change of controllability status of local events induced by actuator failures may jeopardize the performance of the nominal supervisor $S_i (i \in I)$ since a failed actuator may not be enabled/disabled as desired. As a result, our actuator failure tolerance paradigm summons the construction of post-failure DFA model $G_{i}^{F,a}$ of $G_{i} (i \in I)$ in the presence of actuator failures, based on which an appropriate post-failure supervisor $S_{a,i}$ is expected to be synthesized. For such a pursuit, we first introduce the following “suffix automaton” to describe the behavior of a DES after the occurrence of an actuator failure.

**Definition 7:** The suffix automaton after a trim DFA $G = (Q, \Sigma, q_0, \delta, Q_m)$ has generated a word $t \in L(G)$ is another DFA $G^{su, f}(t) = (Q^{su, f}, \Sigma, q_0^{su, f}, \delta^{su, f}, Q_m^{su, f})$, where $q_0^{su, f} = \delta(q_0, t)$, $Q_m^{su, f} = Q_m \cap Q^{su, f}$ with

$Q^{su, f} = \{ q \in Q | (\exists s \in L(G) \setminus \{t\}) [ q = \delta(q_0^{su, f}, s) ] \}$,

and

$(\forall q \in Q^{su, f}, \sigma \in \Sigma) [ \delta^{su, f}(q, \sigma) \iff \delta(q, \sigma) \in Q^{su, f} ]$.

**Example 2:** We now use an example to show the construction of a suffix automaton of a given system. Consider a DES $G_i$ for some $i \in I$, which can be represented by the DFA shown in Fig. 5.

![Fig. 5: The DFA model of $G_i$.](image)

In this example, we assume that $G_i$ has generated a word $t = abac$; and in this case, $G_i$ stays at the state $q_2$. Therefore, the initial state of the suffix automaton of $G_i$ after the generation of $t$, namely, $G_i^{su, f}(abac)$, is $q_2$. The DFA model of $G_i^{su, f}(abac)$ is illustrated in Fig. 6.

![Fig. 6: The suffix automaton $G_i^{su, f}(abac)$.](image)

Intuitively, the suffix automaton $G^{su, f}(s)$ preserves all the possible successive behaviors following the generation of $s \in L(G)$. Thus, it can be readily verified that $L(G^{su, f}) = L(G) \setminus \{s\}$. 

We now proceed to the construction of the DFA models of each subsystem \( G_i(i \in I) \) before and after the occurrence of an actuator failure. Towards this end, we assume that the set \( \Sigma_{i,a} \subset \Sigma_{i,c}(i \in I) \) admits \( K_i \) unreliable actuators, i.e.,

\[
\Sigma_{i,a} = \{ \eta_{i,1}, \eta_{i,2}, \ldots, \eta_{i,K_i} \}.
\]

The presence of \( \Sigma_{i,a} \) introduces a total of \( K_i \) potential failure modes to \( G_i(i \in I) \). We associate each faulty mode \( k \in \{1, 2, \ldots, K_i\} \) with an actuator failure flag event, i.e.,

\[
\Sigma_{i,f}^f = \{ h_{i,1}, h_{i,2}, \ldots, h_{i,K_i} \}
\]

indicating the transition from the nominal mode of \( G_i(i \in I) \) to the failure mode corresponding to the failed actuator \( \eta_{i,k} \). Note that in the \( k \)-th failure mode, \( \eta_{i,k} \) becomes (locally) uncontrollable.

**B. Active Actuator Failure Tolerant Supervisory Control**

On the basis of the previously established notations related to \( G_i(i \in I) \) in the presence of actuator failures, we now derive an automaton-theoretic characterization of the subsystem \( G_i(i \in I) \) after the occurrence of potential actuator failures.

For simplicity of presentation, we assume that an actuator failure can only occur in the nominal mode. Specifically, we assume that the controlled subsystem \( G_i^0(i \in I) \) has generated a word \( t \in L_i^0 \) in the nominal mode before an actuator \( \eta_{i,k}(k = 1, 2, \ldots, K_i) \) fails. In our proposed active failure tolerant supervisory control scheme, the actuator failure flag event \( h_{i,k} \) is then released to interrupt the operation of the nominal supervisor \( S_i \) to avoid undesired behaviors. The post-failure uncontrolled subsystem, denoted as \( F_{i,a}(G_{i,suf}(t)) \), can be viewed as the suffix automaton \( G_{i,suf}^s(t) \) of \( G_i(i \in I) \), except that the local event set \( \Sigma_{i,a}^F \) after the occurrence of \( \eta_{i,k} \) shall be partitioned as

\[
\Sigma_{i,c}^F = \Sigma_{i,c} - \{ \eta_{i,k} \}, \quad \Sigma_{i,uc}^F = \Sigma_{i,uc} \cup \{ \eta_{i,k} \}.
\] (9)

Facing \( F_{i,a}(G_{i,suf}(t)) \), we switch to a post-failure supervisor \( S_i^a \) that is capable of assuring the behavior of the subsystem within the post-failure specification \( L_i' = L_i^0 \setminus \{ t \} \). The overall supervisor switching mechanism is illustrated in Fig. 7.

---

**Fig. 7: Actuator failure tolerant supervision of \( G_i \).**
We apply the $L^*_{{\mathcal{L}$$_G}}$ algorithm for $L'_i$ and $F^k_{i,a}(G^{{\sf suf}}_i(t))$ and the post-failure supervisor $S^a_i$ is synthesized accordingly such that

$$L \left( S^a_i \parallel | F^k_{i,a}(G^{{\sf suf}}_i(t)) \right) = \sup C^k_i(L'_i),$$

where $\sup C^k_i(L'_i)$ is the supremal controllable sublanguage of $L'_i$ with respect to $\Sigma_{i,uc}$ \cite{9}.

We slightly abuse the notations and use $L(S^a_i \parallel | G^F_i,a)$ to denote the closed-loop behavior of $G_i(i \in I)$ in the presence of actuator failures from $\Sigma_{i,a}$. The following theorem states that the switching from the nominal supervisor to the post-failure supervisor will not lead to violation of the local behavior $L^0_i$.

**Theorem 6:** For subsystem $G_i(i \in I)$ with possible actuator failures $\Sigma_{i,a} \subseteq \Sigma_{i,c}$, $L(S^a_i \parallel | G^F_i,a) \models L^0_i$.

**Proof:** As shown in Fig. 7, the closed-loop behavior $L(S^a_i \parallel | G^F_i,a)$ of $G_i(i \in I)$ can be viewed as the sequential concatenation of three parts: i) the nominal behavior $L(S_i \parallel | G_i)$ until the occurrence of the actuator fault; ii) the interruption command modeled by the corresponding actuator fault flag event; iii) the post-fault controlled behavior after we switch to the post-failure supervisor $S^a_i$. We assume without loss of generality that an actuator fault $\eta_{i,k}$ occurs after $G_i(i \in I)$ has executed a word $t \in L^0_i$. According to the previous discussion, in the nominal mode, we can write that

$$\mathcal{T} \subseteq L(S_i \parallel | G_i) = L^0_i.$$

Next, following up the generation of the actuator fault flag event $h_{i,k}$, we interrupt the operation of $S_i$ and aim to synthesize $S^a_i$ that can tolerate the actuator fault $\eta_{i,k}$. From \cite{10}, any behavior $t'$ generated by $F^k_{i,a}(G^{{\sf suf}}_i(t))$ under the supervision of $S^a_i$ can be written as

$$\mathcal{T} \subseteq L \left( S^a_i \parallel | F^k_{i,a}(G^{{\sf suf}}_i(t)) \right) = \sup C^k_i(L'_i) = \sup C^k_i(L^0_i \setminus \{t\}).$$

In summary, the overall behavior performed by $S^a_i \parallel | G^F_i,a(i \in I)$ is formed by the concatenation of $G_i$’s closed-loop behaviors before and after the actuator fault $\eta_{i,k}$:

$$L(S^a_i \parallel | G^F_i,a) = \{ \mathcal{P} \mid t \in L^0_i, t' \in L^0_i \setminus \{t\} \}.$$  

Since $h_{i,k} \not\in \Sigma_i(i \in I)$, it follows from Definition \cite{5} that

$$P_{h_{i-k}}[L(S^a_i \parallel | G^F_i,a)] = \{ \mathcal{P} \mid t \in L^0_i, t' \in L^0_i \setminus \{t\} \} \subseteq \mathcal{T} = L^0_i,$$

with $P_{h_{i-k}}$ being the natural projection from $(\Sigma_i \cup \{h_{i,k}\})^* \to \Sigma_i^*$, which is equivalent to $L(S^a_i \parallel | G^F_i,a) \models L^0_i$. The proof is completed.

**VII. SUPervisory CONTROL IN THE PRESENCE OF MULTIPLE SENSOR AND ACTUATOR FAILURES**

In the previous two sections, we developed failure tolerant control schemes for sensor and actuator failures in a subsystem of the distributed DES, respectively. The proposed results are established under the assumption that only one sensor/actuator failure shall occur in the nominal mode of a subsystem. Nevertheless, multiple failures may occur in a subsystem of the distributed DES. Therefore, we provide some insights on how to enhance the failure tolerance of these subsystems when multiple failures take place. In the rest of this section, three circumstances are
taken into consideration: (i) multiple sensor failures; (ii) multiple actuator failures; and (iii) multiple sensor and actuator failures.

A. Synthesis of Supervisors to Tolerate Multiple Sensor Failures

We start with a nominally supervised subsystem $G_i^0$ ($i \in I$) that may suffer from multiple sensor failures. As discussed in Section V A, the presence of $\Sigma_{i,s}$ induces $N_i$ failure modes in addition to the nominal mode $G_i^0$, each of which is captured by the failure mode DFA

$$G_i^k = (Q_i^k, \Sigma_i, \delta_i^k), \quad k = 1, 2, \ldots, N_i.$$  

In addition to the set of sensor failure flag events $\Sigma_{i,s}^f$, we introduce another set of mode-switching events

$$\Sigma_{i,s}^{sw} = \{f_i^{k_1,k_2} : k_1, k_2 = 1, 2, \ldots, N_i\}$$

to indicate the switch from one sensor failure mode $k_1$ to another mode $k_2$. Similar to $\Sigma_{i,s}^f$, $f_i^{k_1,k_2} \in \Sigma_{i,s}^{sw}$ also denotes an associated mapping

$$f_i^{k_1,k_2} : Q_i^{k_1} \rightarrow Q_i^{k_2}, \quad k_1, k_2 = 1, 2, \ldots, N_i,$$  

which describes the circumstance that the sensor $\sigma_{i,k_2}$ fails in the failure mode $G_i^{k_1}$. In particular, the mapping $f_i^{k_1,k_2}$ satisfies that $f_i^{k_1,k_2}(q_i^{k_1}) = q_i^{k_2}$, which implies that the sensor $\sigma_{i,k_2}$ fails when $G_i^{k_1}$ evolves to state $q_i^{k_1}$. In this case, the initial state of $G_i^{k_2}$ should be $q_i^{k_2}$.

We now consider a closed-loop subsystem $G_i^0$ subject to multiple sensor failures. Without loss of generality, we assume that $J_i$ sensors $\sigma_{i,1}, \sigma_{i,2}, \ldots, \sigma_{i,J_i}$ fail in $G_i$ in a cascaded manner. Similar to the unified DFA $\overline{G}_i$ that is defined in the single-failure case, the failure-pruned behavior of $G_i$ controlled by the nominal supervisor $S_i$ can be characterized by the following trim DFA

$$\overline{G}_i = (\overline{Q}_i, \overline{\Sigma}_i, \overline{q}_i^0, \overline{q}_i),$$

where:

- $\overline{Q}_i = Q_i^0 \cup (\cup_{j=s}^{J_i} Q_i^j)$,
- $\overline{\Sigma}_i = \Sigma_i \cup \Sigma_{i,s}^f \cup \Sigma_{i,s}^{sw}$,
- and

$$\overline{q}_i = (\cup_{j=s}^{J_i} q_i^j) \cup \{(q_i^j, f_i,k,q_i^{k+1}) : f_i,k(q_i^{k+1}) = q_i^{k+1} \}$$

with $j = 1, 2, \ldots, J_i - 1$.

The failure tolerance framework shown in Fig. 2 can be incrementally applied to address multiple sensor failures.

1) When no sensor fails in $G_i$, we use the nominal supervisor $S_i$ such that $L(G_i^0) = L_i^0$.

2) We assume that the sensor $\sigma_{i,1}$ fails when $G_i^0$ generates a word $t_0 \in L_i^0$. In this case, the sensor failure flag event $f_{i,1}$ is generated. We then follow the sensor failure tolerant control strategy shown in Fig. 2 and design the corresponding post-failure supervisor $S_i^{1,s}$ for the post-failure specification $L_i^1 = L_i^0 \setminus \{t_0\}$.  


3) For \( j = 2, 3, \ldots, J_i \), we assume that the sensor \( \sigma_{i,j} \) fails when the underlying subsystem generates a word \( t_{j-1} \in L_i^{j-1} \). In this regard, the mode-switching event \( f_i^{j-1,j} \) is released by the system. The sensor failure tolerant control framework shown in Fig. 2 is applied to synthesize an additional post-failure supervisor \( S_{i,j} \) for the newly-set post-failure specification \( L_i^j = L_i^{j-1} \setminus \{t_{j-1}\} \).

We summarize the incremental synthesis procedure as the following \( SYN - MS \) (which is short for “synthesis for multiple sensor failures”) algorithm.

Algorithm 2: The \( SYN - MS \) Algorithm

**Input:** \( L_i, \Sigma_{i,s}, \Sigma_{i,c} \text{ and } \Sigma_{i,uc} \)

**Output:** \( S_i, S_{i,j}^j \) with \( j = 1, 2, \ldots, J_i \)

1. synthesize \( S_i \leftarrow L_i^*LS(L_i, \Sigma_{i,c}, \Sigma_{i,uc}) \)
2. \( j \leftarrow 1 \)
3. \( L_i^1 \leftarrow L_i^0 \setminus \{I_0\} \)
4. \( \Sigma_{i,c}^{F,s} \leftarrow \Sigma_{i,c}, \Sigma_{i,uc}^{F,s} \leftarrow \Sigma_{i,uc} \cup \{f_{i,1}\} \)
5. \( \Sigma_{i,o}^{F,s} \leftarrow (\Sigma_i - \{\sigma_{i,1}\}) \cup \{\sigma_{i,1}\}, \Sigma_{i,uo}^{F,s} \leftarrow \{f_{i,1}\} \)
6. synthesize \( S_{i,j}^1 \) for \( L_i^1 \) using the \( L_i^*LS \) algorithm, \( \Sigma_{i,o}^{F,s} \text{ and } \Sigma_{i,uo}^{F,s} \)
7. for \( j = 2 \text{ to } J_i \) do
8. \( L_i^j \leftarrow L_i^{j-1} \setminus \{t_{j-1}\} \)
9. \( \Sigma_{i,c}^{F,s} \leftarrow \Sigma_{i,c}, \Sigma_{i,uc}^{F,s} \leftarrow \Sigma_{i,uc} \cup \{f_i^{j-1,j}\} \)
10. \( \Sigma_{i,o}^{F,s} \leftarrow \Sigma_{i,o} \cup \{f_i^{j-1,j}\}, \Sigma_{i,uo}^{F,s} \leftarrow \Sigma_{i,uo} \cup \{\sigma_{i,j}\} \)
11. synthesize \( S_{i,j}^j \) for \( L_i^j \) using the \( L_i^*LS \) algorithm, \( \Sigma_{i,o}^{F,s} \text{ and } \Sigma_{i,uo}^{F,s} \)
12. end for
13. return \( S_i \) and \( S_{i,j}^j \) with \( j = 1, 2, \ldots, J_i \)

We slightly abuse the notations and denote \( L(S_i^*||G_i^{F,s}) \) as the closed-loop behavior of \( G_i \) \((i \in I)\) in the presence of the failed sensors \( \sigma_{i,1}, \sigma_{i,2}, \ldots, \sigma_{i,J_i} \) and their corresponding failure tolerant supervisors \( S_{i,1}^{s}, S_{i,2}^{s}, \ldots, S_{i,J_i}^{s} \).

From the computation procedure of the \( SYN - MS \) algorithm, one can conclude that the closed-loop behavior of the subsystem \( G_i \) \((i \in I)\) shall be of the following form.

\[
L(S_i^*||G_i^{F,s}) = \left\{ t_0 f_{i,1} t_1 f_{i,1}^{1.2} t_2 \cdots t_{J_i-1} f_{i,J_i-1} f_i^{J_i-1,J_i} t_i \right\},
\]

where \( t_0 \in L_i^0 \), and \( t_j \in L_i^{j-1} \setminus \{t_{j-1}\} \) for \( j = 1, 2, \ldots, J_i \). It then follows from the proof of Theorem 6 that the \( SYN - MS \) algorithm can properly handle the multiple-failure case.

**Theorem 7:** For the subsystem \( G_i \) \((i \in I)\) that is affected by sensor failures \( \sigma_{i,1}, \sigma_{i,2}, \ldots, \sigma_{i,J_i} \), the nominal supervisor \( S_i \) and the sensor failure tolerant supervisors \( S_{i,1}^{s}, S_{i,2}^{s}, \ldots, S_{i,J_i}^{s} \) that are synthesized via the \( SYN - MS \) algorithm will jointly enforce the fulfillment of the assigned local specification \( L_i \), i.e., \( L(S_i^*||G_i^{F,s}) \models L_i^0 \subseteq L_i \).
B. Actuator Failure Tolerant Control against Multiple Failures

Next, we focus on the failure tolerance of local supervisor policies when multiple actuator failures take place. In this regard, we also introduce the set of mode-switching events among different failure modes when multiple actuators may fail. Formally, for the subsystem $G_i$ that may admit $K_i$ potential actuator failures, the set of mode-switching events is given by

$$\Sigma_{i,a}^{sw} = \{h_i^{m_1,m_2} : m_1, m_2 = 1, 2, \ldots, K_i\}.$$  

Intuitively, an event $h_i^{m_1,m_2}$ is generated when actuator $\eta_{i,m_2}$ fails after $\eta_{i,m_1}$ has already failed in $G_i$.

Consider the case in which $G_i$ suffers from subsequent occurrence of $Z_i$ actuator failures, namely, $\eta_{i,1}, \eta_{i,2}, \ldots, \eta_{i,Z_i}$. The supervisor switching mechanism illustrated in Fig. [7] is inherited here to solve the failure tolerant supervisory control problem.

1) When no actuator failure occurs in $G_i$, we use the nominal supervisor $S_i$ to assure $L(G_i^0) = L_i^0$. 

2) Assume that actuator $\eta_{i,1}$ fails after $G_i^0$ generates a word $t_0 \in L_i^0$. The actuator failure flag event $h_{i,1}$ is then generated to interrupt the operation of $S_i$. The post-failure uncontrolled system is then replaced by $G_i = F_i^{1,0}(G_i^{su})(t_0)$ with appropriate post-failure local events. We employ the $L_i^1$ algorithm to synthesize the corresponding post-failure supervisor $S_i^{1,a}$ for the post-failure specification $L_i^1 = L_i^0 \setminus \{t_0\}$.

3) For $m = 2, 3, \ldots, Z_i$, the underlying subsystem generates a word $t_{m-1} \in L_i^{m-1}$ when a subsequent actuator failure $\eta_{i,m}$ occurs. Meanwhile, the associated mode-switching event $h_i^{m-1,m}$ is released. The actuator failure tolerant control framework shown in Fig. [7] is applied to synthesize a new post-failure supervisor $S_i^{m,a}$ for the post-failure uncontrolled subsystem $G_i = F_i^{m,0}(G_i^{su})(t_{m-1})$ and the post-failure specification $L_i^m = L_i^{m-1} \setminus \{t_{m-1}\}$.

The actuator failure tolerant control schemes stated above can be summarized as the following $\text{SYN} - \text{MA}$ (which stands for “synthesis for multiple actuator failures”) algorithm.

We also use $L(S_i^a \mid G_i^{F,a})$ to denote the closed-loop behavior of $G_i$ ($i \in I$) in the presence of the failed actuators $\eta_{i,1}, \eta_{i,2}, \ldots, \eta_{i,Z_i}$ and control policies issued by $S_i^{1,a}, S_i^{2,a}, \ldots, S_i^{Z,a}$ individually in succession. It follows immediately from the $\text{SYN} - \text{MA}$ algorithm that the generated language of the closed-loop subsystem $L(S_i^a \mid G_i^{F,a})$ shall be of the following form.

$$L(S_i^a \mid G_i^{F,a}) = \left\{t_0h_{i,1}t_1h_i^{1,2}t_2\cdots t_{Z_i-1}h_i^{Z_i-1,Z_i}t_{Z_i}\right\},$$

where $t_0 \in L_i^0$, and $t_m \in L_i^{m-1} \setminus \{t_{m-1}\}$ for $m = 1, 2, \ldots, Z_i$. Thus, we can also extend Theorem 5 to cope with multiple actuator failures.

**Theorem 8:** Consider the subsystem $G_i$ ($i \in I$) that is affected by actuator failures $\eta_{i,1}, \eta_{i,2}, \ldots, \eta_{i,Z_i}$. The nominal supervisor $S_i$ and the supervisor switching among $S_i^{1,a}, S_i^{2,a}, \ldots, S_i^{Z,a}$ determined by the $\text{SYN} - \text{MA}$ algorithm shall guarantee the accomplishment of the assigned local specification $L_i$, i.e., $L(S_i^a \mid G_i^{F,a}) = L_i^0 \subseteq L_i$.

C. Failure Tolerant Control of DESs with Multiple Failures

We conclude this section by studying the failure tolerant supervisory control of a subsystem $G_i$ ($i \in I$) that is influenced by both sensor and actuator failures. Without loss of generality, we consider the case in which an
Algorithm 3: The $SYN - MA$ Algorithm

**Input:** $L_i, \Sigma_{i,s}, \Sigma_{i,c}$ and $\Sigma_{i,uc}$

**Output:** $S_i, S_i^{m,a}$ with $m = 1, 2, \ldots, Z_i$

1. synthesize $S_i \leftarrow L_{LS}^*(L_i, \Sigma_{i,c}, \Sigma_{i,uc})$
2. $m \leftarrow 1$
3. Stop the operation of $S_i$
4. $L_i^1 \leftarrow L_i^0 \setminus \{t_0\}$
5. $G_i \leftarrow F_{i,a}^1(G_{i,uf}^s(t_0))$
6. $\Sigma_{i,c}^{F,a} \leftarrow \Sigma_{i,c} - \{\eta_i,1\}$, $\Sigma_{i,uc}^{F,a} \leftarrow \Sigma_{i,uc} \cup \{h_i,1\}$
7. synthesize $S_i^{1,a} \leftarrow L_{LS}^*(L_i^1, \Sigma_{i,c}^{F,a}, \Sigma_{i,uc}^{F,a})$
8. for $m = 2$ to $Z_i$ do
9. Stop the operation of $S_i^{m-1,a}$
10. $L_i^m \leftarrow L_i^{m-1} \setminus \{t_{m-1}\}$
11. $G_i \leftarrow F_{i,a}^m(G_{i,uf}^s(t_{m-1}))$
12. $\Sigma_{i,c}^{F,a} \leftarrow \Sigma_{i,c} - \{\eta_i,m\}$, $\Sigma_{i,uc}^{F,a} = \Sigma_{i,uc}^{F,a} \cup \{h_i,m-1,m\}$
13. synthesize $S_i^{m,a} \leftarrow L_{LS}^*(L_i^m, \Sigma_{i,c}^{F,a}, \Sigma_{i,uc}^{F,a})$
14. end for
15. return $S_i$ and $S_i^{m,a}$ with $m = 1, 2, \ldots, Z_i$

actuator $\eta_{i,m_1}$ fails first, and a sensor failure $\sigma_{i,k_1}$ occurs afterwards. The failure tolerance of the supervisory control strategy of $G_i$ is then achieved as follows.

1) When no failure is detected in $G_i$, the nominal supervisor $S_i$ is employed to guarantee $L(G_i^0) = L_i^0$.

2) When the actuator failure $\eta_{i,m_1}$ is detected after $G_i^0$ generates a word $t_0$, the subsystem generates $h_{i,m_1}$ to interrupt the operation of $S_i$. Facing the post-failure uncontrolled subsystem $G_i = F_{i,a}^{m_1}(G_{i,uf}^s(t_0))$ and the post-failure specification $L_i^{m_1} = L_i^0 \setminus \{t_0\}$, we apply the $SYN - MA$ algorithm and switch to the post-failure supervisor $S_i^{m_1,a}$. The new closed-loop system and behavior are still denoted as $G_i^0$ and $L_i^0$, respectively.

3) When the sensor $\sigma_{i,k_1}$ fails while the new $G_i^0$ generates a word $t_1$, the sensor failure flag event corresponding to $\sigma_{i,k_1}$, $f_{i,k_1}$, is issued. We deploy the $SYN - MS$ algorithm to design the corresponding failure tolerant supervisor $S_i^{k_1,s}$ for the post-failure specification $L_i^1 = L_i^0 \setminus \{t_1\}$.

To pursue concise notations, we use $S_i^F||G_i^F$ as a unified notation to represent the closed-loop model of the subsystem $G_i$ ($i \in I$) in the presence of failures in the rest of this paper. The following theorem, which asserts that the integration of the failure tolerant control algorithms will not lead the subsystem to exceed its local specification, can be viewed as an immediate result by combining Theorem 5 and Theorem 6.

**Theorem 9:** Consider the subsystem $G_i$ ($i \in I$) that is subject to an actuator failure $\eta_{i,m_1}$ and a sensor failure $\sigma_{i,k_1}$. The established integration of the $SYN - MA$ and $SYN - MS$ algorithms results in $L(S_i^F||G_i^F) \Rightarrow L_i^0 \subseteq L_i$.

**Remark 5:** The extension of Theorem 5 to multiple sensor and actuator failures that occur in arbitrary orders can
also be achieved (subject to the orders of applying the SYN – MS and SYN – MA algorithms).

VIII. FAILURE TOLERANT COORDINATION OF DISTRIBUTED DESs VIA COMPOSITIONAL VERIFICATION

This section employs a compositional verification procedure to determine whether or not the collective behaviors of all the controlled subsystems satisfy the global specification $L$ in the presence of potential sensor and/or actuator failures.

A. Compositional Verification via Assume-guarantee Reasoning

We first briefly review the compositional verification techniques. Let $G$ be a DFA accepting a regular language $L$, the complete DFA of $G$ is defined as follows.

**Definition 8 (Complete DFA):** Let $q_e$ be an "error state". Given a DFA $G = (Q, \Sigma, q_0, \delta, Q_m)$, the complete model of $G$ is defined as a DFA $\tilde{G} = (\tilde{Q}, \Sigma, q_0, \tilde{\delta}, \{q_e\})$, where $\tilde{Q} = Q \cup \{q_e\}$, and

$$\forall \tilde{q} \in \tilde{Q}, \sigma \in \Sigma, \tilde{\delta}(\tilde{q}, \sigma) = \begin{cases} \delta(\tilde{q}, \sigma), & \text{if } \tilde{q} \in Q \land \delta(\tilde{q}, \sigma) \neq \emptyset, \\ q_e, & \text{if } \tilde{q} = q_e \lor \delta(\tilde{q}, \sigma) = \emptyset. \end{cases}$$

It is clear that $L_m(\tilde{G}) = \Sigma^*$. Let $coG$ denote a DFA that accepts the complement of $L$, i.e., $L_m(coG) = \Sigma^* - L$. It can be verified that $coG$ is constructed by swapping the marked states of $\tilde{G}$ with its non-marked states and vice versa, i.e., $coG = (\tilde{Q}, \Sigma, q_0, \tilde{\delta}, \tilde{Q} - Q_m)$.

The idea of assume-guarantee reasoning paradigm [30] is leveraged to enhance the scalability of compositional verification. An assume-guarantee formula is a triple $\langle A \rangle M \langle P \rangle$, where $M$ is a system model, $P$ is a property to be verified and $A$ is an assumption about $M$’s environment, all of which are represented by an associated DFA. The formula holds if whenever $M$ is part of a system satisfying $A$, the system must also guarantee the property $P$, i.e., $\forall E, E || M = A \implies E || M = P$ [30].

If $M$ is composed of multiple components, i.e., $M = \big||_{i \in I} M_i$, the following symmetric proof rule, namely SYM-N [30], is applied for the compositional verification.

$$1 \langle A_1 \rangle M_1 \langle P \rangle$$

$$\ldots$$

$$n \langle A_n \rangle M_n \langle P \rangle$$

$$n + 1 L(coA_1||coA_2||\cdots||coA_n) \subseteq L(P)$$

$$\langle true \rangle (M_1||M_2||\cdots||M_n) \langle P \rangle$$

where $A_i$ is an assumption about the component $M_i$’s environment and $coA_i$ is the complement DFA of $A_i$. The concept of weakest assumption is a key ingredient of compositional verification, which is defined formally as follows.

**Definition 9 (Weakest Assumption for $\Sigma_i,IF$):** (Adapted from [30]) Let $M = \big||_{i \in I} M_i$ be a distributed system with local alphabets $\Sigma_i (i \in I)$, $P$ be a DFA representation of a property and $\Sigma_i,IF$ be a specified interface alphabet of $M_i$ to its environment. The weakest assumption $A^{we,\Sigma_i,IF}_i$ of $M_i$ over the alphabet $\Sigma_i,IF$ for property $P$ is a DFA
such that: (i) $\Sigma(A_i^{w,\Sigma_i,IF}) = \Sigma_i,IF$; and (ii) for any $M_{i-j} = \bigcup_{i,j \neq i} M_j$, $\langle true \rangle M_i \langle \langle (P_i,IF)(M_{i-j}) \rangle \rangle$ if and only if $\langle true \rangle M_{i-j} \langle \langle A_i^{w,\Sigma_i,IF} \rangle \rangle$, where $P_i,IF$ is the natural projection from $\bigcup_{i \in I} \Sigma_i$ to $\Sigma_i,IF$.

Throughout the rest of this paper, we require that $\Sigma_P \subseteq \bigcup_{i \in I} \Sigma_i$, and we use the notation $A_i^{w}$ to denote the weakest assumption for component $M_i$ over $\Sigma_i,IF$ being set to $\Sigma_A$, such that $\Sigma_A \subseteq (\bigcap_{i \in I} \Sigma_i) \cup \Sigma_P(i \in I)$.

**B. Compositional Verification and Counterexample-guided Re-synthesis**

We start with the compositional verification of the controlled system in the nominal mode. To apply the SYM-N proof rule for compositional verification, we let $M_i := S_i | G_i$ be the component module of the supervised subsystem $G_i$ ($i \in I$) in the nominal mode. When no failures are detected, we need to determine whether or not $M \models L$, where $M = \bigcup_{i \in I} M_i$ denotes the collective behavior performed by the controlled distributed DES.

For the purpose of assume-guarantee reasoning, we propose another modification of the $L^*$ algorithm, namely the $L^*_CV$ algorithm, to learn appropriate and weakest local assumptions $A_i$ for each module $M_i$ in the nominal mode or $M_i^F$ ($i \in I$) in the failure mode.

With slightly abusing the notations, we use $L$ and $coL$ to denote the DFA that recognizes $L$ and its complement language, respectively. A two-layered implementation of the $L^*_CV$ algorithm via the SYM-N proof rule is depicted in Fig. 8. In the first layer, the $L^*_CV$ algorithm associates $n$ local Teachers with $M_i$ ($i \in I$), to learn the appropriate and weakest assumption $A_i$ for $M_i$. As the $L^*_CV$ algorithm executes the learning procedure iteratively, it incrementally (in the sense of the number of states) constructs a sequence of assumption DFAs $\{A_i^j\}_{j \in \mathbb{N}}$ for $M_i$, which in turn converges to the DFA that recognizes the weakest assumption $A_i^w$ whose event set is constrained to be $\Sigma_A$.

The implementation of the $L^*_CV$ algorithm relies on the following lemma.

**Lemma 1:** (Adapted from [30]) Given $t \in \Sigma^*$, $i \in I$ and a property $L$, $t \in L(A_i^w)$ if and only if $\langle DF_A(t) \rangle M_i \langle L \rangle$ holds, where $DF_A(t)$ is a trim DFA such that $L(DF_A(t)) = L_{m}(DF_A(t)) = \overline{t}$.

The subscript “CV” stands for “compositional verification”.

---

**Fig. 8:** Compositional verification by learning assumptions.
Inspired by Lemma 1, each local Teacher of the \( L^*_{CV} \) algorithm answers the following membership queries to construct local assumption DFAs for each component module \( M_i \): for \( i \in I \) and \( t \in \Sigma^* \),

\[
T_i(t) = \begin{cases} 
1, & \text{if } \langle DFA(t) \rangle M_i \langle L \rangle \text{ is true}, \\
0, & \text{otherwise.} 
\end{cases} 
\] (11)

In addition to the membership queries (11), the Teacher of the \( L^*_{CV} \) algorithm justifies the conjecture \( \langle A^i_t \rangle M_i \langle L \rangle \) rather than \( L(A^i_t) = L(A^i_u) \). Once the Teacher denies the conjecture, a counterexample \( t \in \Sigma^* \) is proposed by the Teacher, and the \( L^*_{CV} \) algorithm adds \( P_{A_i}(t) \) and all its prefixes back to the local iteration loop to update the observation tables, where \( P_{A_i} \) is the natural projection from \( \Sigma^* \) to \( \Sigma^*_{A_i} \).

Once the local learning loop of the \( L^*_{CV} \) algorithm terminates, we collect a family of local assumptions \( A_i := A_i^w (i \in I) \). The \( L^*_{CV} \) algorithm proceeds to the second layer and deploys Teacher \( n + 1 \) to justify whether \( L(coA_1 \| \cdots \| coA_n) \subseteq L \) or not. If Teacher \( n + 1 \) returns “True”, the synthesis framework terminates with the conclusion that \( \|_{i \in I} M_i \models L \). Otherwise, Teacher \( n + 1 \) returns “False” with a counterexample \( t \in \Sigma^* \). The \( L^*_{CV} \) algorithm then determines whether or not the global mission \( L \) is indeed violated by the collective behavior of \( M \), which is performed by simulating \( t \) on each composed DFA \( M_i \| coL (i \in I) \) and by checking whether or not \( t \) can be accepted. If \( t \) is not a violating trace for at least one system component \( M_i \), we treat \( t \) in the same way in the first layer and use \( t \) to reconstruct the local assumption \( A_i \) for \( M_i \). Otherwise, \( t \) turns out to be a common violating word of all subsystems and the controlled distributed DES \( \|_{i \in I} M_i \) indeed violates \( L \). As shown in Fig. 1, undesired mission behaviors emerge in the joint execution of synthesized mission plans \( L^\text{mi}_i \) for \( G_i (i \in I) \); therefore, the re-synthesis of local mission plans is triggered (cf. Section III.C). The correctness and termination properties of the \( L^*_{CV} \) algorithm are summarized in the following theorem.

**Theorem 10:** For the global specification \( L \) and the controlled system components \( M_1, M_2, \ldots, M_n \), the \( L^*_{CV} \) algorithm implemented by the framework shown in Fig. 8 with the SYM-N proof rule terminates within finite number of iterations and correctly returns whether or not \( \|_{i \in I} M_i \models L \).

The counterexample generated by the compositional verification procedure is further utilized to guide the re-synthesis of local (nominal) supervisors. Whenever a violating counterexample \( t \in \Sigma^* \) is presented by the compositional verification procedure, the re-synthesis is accomplished by the following steps: first, let

\[
L^\text{temp}_i := L_i - P_i(t); 
\] (12)

next, we provide \( G_i (i \in I) \) with the following new local specification

\[
L_i := L^\text{temp}_i - coL^\text{temp}_i \Sigma_i^*, 
\] (13)

where \( coL^\text{temp}_i = \Sigma_i^* - L^\text{temp}_i \) is the complement language of \( L^\text{temp}_i \). It follows from 31 that the obtained \( L_i (i \in I) \) is the supremal prefix-closed sublanguage of \( L^\text{temp}_i \); therefore it can be used as a locally feasible specification. The \( L^*_{LS} \) algorithm is further applied to synthesize the new local supervisor \( S_i \).

Since it has been shown in the previous subsections that both the \( L^*_{LS} \) and \( L^*_{CV} \) algorithms possess finite convergence, we mainly focus on evaluating the performance of the proposed coordination and control schemes.
We introduce the notion of separate controllability to characterize the solutions of the coordination and control of distributed DESs.

**Definition 10:** Given the distributed DES $G$ that consists of $n$ subsystems $G_i$ whose locally controllable events are included in $\Sigma_{i,c} \subseteq \Sigma_i$, $i \in I$, a prefix-closed language $L \subseteq \Sigma^*$ is said to be separately controllable with respect to $\Sigma_i$, $\Sigma_{i,uc}$ and $G_i$ if: i) $L = \bigcup_{i \in I} P_i(L)$; ii) $P_i(L)$ is controllable with respect to $G_i$ and $\Sigma_{i,uc}$.

The following theorem states that the separate controllability is a sufficient and necessary condition for the existence of appropriate local supervisors.

**Theorem 11:** [34] Given the distributed DES $G$ that consists of $n$ subsystems $G_i (i \in I)$ with $\Sigma_{i,c}$ and $\Sigma_{i,uc}$, and a non-empty and prefix-closed global specification $L \subseteq L(G)$, there exists a family of local supervisors $S_i (i \in I)$ for $G_i$ such that $L(S_i || G_i) = P_i(L)$ and $||_{i \in I} L(S_i || G_i) = L$ if and only if $L$ is separately controllable with respect to $\Sigma_i$, $\Sigma_{i,uc}$ and $G_i$.

Based on the aforementioned properties of separately controllable languages, the following theorem is established to illustrate the correctness of the proposed coordination and control scheme of the distributed DESs in the nominal case.

**Theorem 12:** [34] The distributed coordination and control framework in Fig. 1 terminates and returns local nominal supervisors $S_i$ as well as local specifications $L_i (i \in I)$ for each subsystem such that the collective behaviors of the distributed DES $G$ achieve a separately controllable sublanguage of $L$, within a finite number of iterations.

**C. Coordination of Distributed DESs with Failures**

When nominal evolution of subsystems of the distributed system $G$ is affected by sensor and/or actuator failures, the coordination of various subsystems needs to be reorganized such that the global specification will not violated. In this subsection, we study how the compositional verification and counterexample-guided supervisor re-synthesis can be applied to the distributed DESs in the presence of potential failures.

Without loss of generality, we assume that during the nominal operation of the system $G$, a subsystem $G_i$ suffers from sensor and actuator failures, for some $i \in I$. In this regard, whenever sensor and/or actuator failures are detected in $G_i$, we activate the associated failure tolerant supervisory control reconfiguration procedures to resolve the influence of the failures. On the one hand, it turns out that, according to Theorem 9, such supervisory control strategies will not lead to falsification of the assigned local specification. On the other hand, the failure flag event sets $\Sigma_{i,s}^F$ and $\Sigma_{i,a}^F$ that indicate the transition from the nominal mode to a failure mode, as well as the mode-switching event sets $\Sigma_{i,s}^{sw}$ and $\Sigma_{i,a}^{sw}$ that represent the switching among various failure modes, are all local events of the subsystem $G_i$, which implies that execution of these events will not affect the coordinated performance of other subsystems than $G_i$ according to the definition of parallel composition. Therefore, rather than use $M_i^F := S_i^F || G_i^F$ as the component module of the closed-loop behavior of $G_i$ in the presence of failures, we adopt $M_i^F$ to be a DFA defined over $\Sigma_i^F := \Sigma_i^{sw} \cup \Sigma_i^{sw}$ that satisfies

$$L(M_i^F) = L_m(M_i^F) = P_i,F \left[ L(S_i^F || G_i^F) \right],$$
where \( P_{i,F} \) is defined to be the natural projection from \( \Sigma^F_i \) to \( \Sigma_i \). It is worth pointing out that from the conclusion of Theorem 9 the following language inclusion holds:

\[
L(M^F_i) \subseteq L^0_i \subseteq L_i.
\] (14)

The DFA \( M^F_i \) is updated once occurrence of a new sensor/actuator failure is detected and is presented as the component module of subsystem \( G_i \) to the compositional verification procedure.

When the counterexample \( t \in \Sigma^* \) is generated by the compositional verification, we follow the computation of (12) and (13) to re-assign the new local specification \( L_i \) to \( G_i \). It can be verified that Theorem 12 applies for the failure-pruned subsystem \( G_i \) (hence the distributed system \( G \)) as well.

IX. AN ILLUSTRATIVE EXAMPLE OF ROBOTIC COORDINATION

A. DES Models of the Multi-robot System

We now apply the proposed failure tolerant synthesis framework to a multi-robot coordination experimental scenario [34] to examine its tolerance to potential failures. Consider a cooperative robotic team that consists of three robots, namely \( G_1 \), \( G_2 \) and \( G_3 \), all of which are supposed to possess identical sensing and communication capabilities. Furthermore, we assume that \( G_2 \) has actuators that offer rescue and fire-fighting capabilities. Initially, the three robots are positioned in Room 1. Room 2 and Room 3 are accessible using the one-way door \( D_2 \), or the two-way doors \( D_1 \) and \( D_3 \), respectively. Each door is equipped with a spring and close automatically, whenever there is no force to keep them open. The working environment shared by the three robots is depicted in Fig. 9.

Assume that the fire alarm in Room 2 is triggered. After receiving the fire-extinguishing request, \( G_2 \) needs to go to Room 2 immediately through the one-way door \( D_2 \) to accomplish this task. Afterwards, \( G_2 \) is required to return to Room 1 from the two-way door \( D_1 \). In this example, \( D_1 \) is heavy and needs the collaboration of the robots \( G_1 \) and \( G_3 \) to be opened. After \( G_2 \) returns to Room 1, both \( G_1 \) and \( G_3 \) move backwards to close \( D_1 \) and all three robots assemble in Room 1 for next fire alarm.

To formalize this coordination scenario, we translate the specification into a regular language by introducing the events in Table 1, where \( i, k \in \{1, 2, 3\} \).
| Event     | Explanation                                      |
|-----------|--------------------------------------------------|
| $h_i$     | Robot $G_i$ receives the service request, $i = 1, 2, 3$. |
| $FF$      | Robot $G_2$ extinguishes the fire.               |
| $G_i to D_1$ | Robot $G_i$ approaches the door $D_1$, $i = 1, 3$. |
| $G_i on D_1$ | Robot $G_i$ at the door $D_1$, $i = 1, 3$.       |
| $G_i in k$  | Robot $G_i$ heads for Room $k$.                 |
| $G_i out k$ | Robot $G_i$ stays at Room $k$.                  |
| $Open$    | command for moving forward to open $D_1$.        |
| $Close$   | command for moving backward to close $D_1$.      |
| $D_1 open$ | $D_1$ is opened.                                |
| $D_1 closed$ | $D_1$ is closed.                       |
| $r$       | All the robots return to Room 1.                |

For $i \in \{1, 2, 3\}$, local event sets $\Sigma_i$ are defined as follows:

$$\Sigma_1 = \{h_1, G_1 to D_1, G_1 on D_1, Open, Close, G_2 in 1, G_1 to 3, G_{1in3}, D_{1closed}, D_{1open}, G_1 to 1, G_1 in 1, r \},$$

$$\Sigma_2 = \{h_2, FF, G_2 to 2, G_2 in 2, D_{1open}, G_2 to 1, G_2 in 1, FF, r \},$$

$$\Sigma_3 = \{h_3, G_3 to 3, G_{3in3}, G_3 to D_1, G_3 on D_1, Open, Close, D_{1open}, G_2 in 1, D_{1closed}, G_3 to 1, G_3 in 1, r \}.$$  

For the purpose of supervisory control, we assume that

$$\Sigma_{1, uc} = \{h_1, G_2 in 1, G_{1in3}, G_{1in1} \},$$

$$\Sigma_{2, uc} = \{h_2, G_2 in 2, D_{1open}, G_2 in 1 \},$$

$$\Sigma_{3, uc} = \{h_3, G_{3in3}, G_2 in 1, G_{3in1} \}.$$  

Furthermore, to account for potential sensor and/or actuator failures that may occur in the robots, we assume that the sets of local sensor events associated with each robot are given by

$$\Sigma_{1, s} = \{h_1, G_2 in 1, G_{1in3}, G_{1in1} \},$$

$$\Sigma_{2, s} = \{h_2, G_2 in 2, D_{1open}, G_2 in 1 \},$$

$$\Sigma_{3, s} = \{h_3, G_{3in3}, G_2 in 1, G_{3in1}, r \};$$

whereas the sets of local actuator events are defined as

$$\Sigma_{1, a} = \{G_1 to D_1, Open, Close, G_1 to 3, G_1 to 1, r \},$$

$$\Sigma_{2, a} = \{FF, G_2 to 2, G_2 to 1, r \},$$

$$\Sigma_{3, a} = \{G_3 to 3, G_3 to D_1, Open, Close, G_3 to 1, r \}.$$  

for robot $G_i$, $i = 1, 2, 3$.  

As discussed in the previous subsection, two regular specifications \( L_{spe}^1 \) and \( L_{spe}^2 \) are introduced corresponding to the fire-extinguishing task as well as the open of \( D_1 \):

\[
\begin{align*}
L_{spe}^1 &= (h_2G_2 to 2G_2 in 2FFD_1 open G_2 to 1G_2 in 1r)^*, \\
L_{spe}^2 &= (h_1G_1 to D_1G_1 on D_1 D_1 open D_1 open G_2 in 1Close D_1 closed r)^* \\
&+ (h_1G_1 to 3G_1 in 3G_1 to D_1 G_1 on D_1 Open D_1 open G_2 in 1Close D_1 closed r)^*. \\
&\| (h_2G_2 to 2G_2 in 2FFD_1 open G_2 to 1G_2 in 1r)^* \\
&\| (h_3G_3 to D_1G_3 on D_1 Open D_1 open G_2 in 1Close D_1 closed r)^* \\
&+ (h_3G_3 to 3G_3 in 3G_3 to D_1 G_3 on D_1 Open D_1 open G_2 in 1Close D_1 closed r)^*. \\
&\| (h_2G_2 to 2G_2 in 2FFD_1 open G_2 to 1G_2 in 1r)^*. \\
\end{align*}
\]

The global task requires simultaneous accomplishment of both \( L_{spe}^1 \) and \( L_{spe}^2 \); that is, the global specification shall be written as \( L = L_{spe}^1 \| L_{spe}^2 \). Our design objective is to synthesize local nominal supervisors \( S_i \) when no failure is detected, as well as failure tolerant supervisors \( S_i^F (i = 1, 2, 3) \) when possible failures occur.

**B. Synthesis of Nominal Supervisors**

As suggested by the failure tolerant coordination and control framework shown in Fig. 1, the local specifications \( L_i (i = 1, 2, 3) \) are initially assigned by \( L_i = P_i(L) \). Toward this end, the DFA representations of \( L_i (i = 1, 2, 3) \) are illustrated in Figures 10 and 11, respectively.

![Fig. 10: Local specifications \( L_1 \) and \( L_3 \)](image)

![Fig. 11: Local specification \( L_2 \)](image)

The nominal supervisor \( S_i (i = 1, 2, 3) \) can therefore be synthesized by applying the \( L_{LS}^* \) and \( L_{CV}^* \) algorithms with respect to the assigned \( L_i \)'s. For the multi-robot system, the DFA models of the synthesized local supervisors are shown in Figures 12, 13 and 14, respectively. Intuitively, to jointly achieve the global specification \( L \), we require that on the one hand, Robot \( G_1 \) stays in Room 1 while \( G_3 \) enters Room 3 in an effort to open \( D_1 \); on the other hand, \( G_2 \) first extinguishes the fire in Room 2 and then returns to Room 1 through \( D_1 \).
C. Post-failure Control Reconfiguration

We now proceed to the accommodation and control reconfiguration strategies for each robot in the presence of potential sensor and/or actuator failures. Let $L_3^0$ denote the controlled behavior of Robot $S_3 || G_3$ in the nominal case. It is clear that $L_3^0 = L(S_3)$. We assume that the sensor $G_3in1$ is failed after the behavior $t = h_3G_3to3G_3in3G_3toD_1G_3onD_1open$ has been executed by the nominal supervisor $S_3$ of Robot $G_3$. In this case, we follow the procedure presented in Section V to construct the DFA model $G_3^{G_3in1}$ of the controlled robot subject to the sensor failure $G_3in1$. With the introduction of the sensor failure flag event $f_3,G_3in1$, the post-failure DFA model of $G_3$ is given in Fig. 15.

Fig. 15: The controlled robot $G_3^{G_3in1}$ in the presence of the failed sensor.

We then apply the proposed sensor failure tolerant control strategy to the DFA model of $G_3^{G_3in1}$. In this example, $\Sigma_{i,uo} = \{G_3in1\}$, and the post-failure specification is given by $L'_3 = L_3^0 \setminus t$. It then follows from Theorem 5 that
the overall controlled behavior of Robot $G_3$ in the presence of the failed sensor $G_3in1$ can be obtained as

$$L(S^*_3||G_3^F) = \{t \mid t \in L_0^1, t' \in L_3^1\}$$

$$= (h_3G_3to3G_3in3G_3toD_1G_3onD_1open \circ f_3,G_3in1C_1toD_1closedG_3to1G_3in1r)^*.$$

X. CONCLUSION AND FUTURE WORK

In this paper, we present a coordination and control framework for distributed DESs in order to tolerate potential sensor and actuator failures. The DES under consideration is modeled as a parallel composition of multiple subsystems, each of which is modeled by a finite automaton. Our proposed framework ensures the accomplishment of the global specification by synthesizing appropriate local supervisors corresponding to each subsystem. Furthermore, automaton-theoretic methods are presented to construct models for the subsystems when either an observable sensor event becomes unobservable or a controllable actuator event turns into uncontrollable. Post-failure supervisors are synthesized accordingly to various failure modes such that the fulfillment of the global specification can still be maintained. The effectiveness of our proposed approach is demonstrated by an illustrative example.

The future work aims to extend the proposed failure tolerant coordination and control framework to deal with additional types of failures, such as general event faults and supervisor failures. Furthermore, coordination and control reconfiguration strategies for distributed DESs under intentional attacks are also expected to be explored.

REFERENCES

[1] A. Teixeira, I. Shames, H. Sandberg, and K. H. Johansson, “A secure control framework for resource-limited adversaries,” *Automatica*, vol. 51, pp. 135–148, 2015.

[2] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*, 2nd ed. Berlin: Springer, 2006.

[3] J. Zaytoon and S. Lafortune, “Overview of fault diagnosis methods for discrete event systems,” *Annu. Rev. Control*, vol. 37, no. 2, pp. 308–320, 2013.

[4] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. New York: Springer, 2008.

[5] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, “Diagnosability of discrete-event systems,” *IEEE Trans. Autom. Control*, vol. 40, no. 9, pp. 1555–1575, 1995.

[6] R. Debouk, S. Lafortune, and D. Teneketzis, “Coordinated decentralized protocols for failure diagnosis of discrete event systems,” *Discrete Event Dynam. Syst.: Theory Applicat.*, vol. 10, no. 1-2, pp. 33–86, 2000.

[7] W. Qiu and R. Kumar, “Decentralized failure diagnosis of discrete event systems,” *IEEE Trans. Syst., Man, Cybern. A, Syst.,Humans*, vol. 36, no. 2, pp. 384–395, 2006.

[8] Y. Wang, T.-S. Yoo, and S. Lafortune, “Diagnosis of discrete event systems using decentralized architectures,” *Discrete Event Dynam. Syst.: Theory Applicat.*, vol. 17, no. 2, pp. 233–263, 2007.

[9] O. Contant, S. Lafortune, and D. Teneketzis, “Diagnosability of discrete event systems with modular structure,” *Discrete Event Dynam. Syst.: Theory Applicat.*, vol. 16, no. 1, pp. 9–37, 2006.

[10] K. W. Schmidt, “Verification of modular diagnosability with local specifications for discrete-event systems,” *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 5, pp. 1130–1140, 2013.

[11] R. Su and W. M. Wonham, “Global and local consistencies in distributed fault diagnosis for discrete-event systems,” *IEEE Trans. Autom. Control*, vol. 50, no. 12, pp. 1923–1935, 2005.

[12] F. Lin, “Robust and adaptive supervisory control of discrete event systems,” *IEEE Trans. Autom. Control*, vol. 38, no. 12, pp. 1848–1852, 1993.

[13] P. J. Ramadge and W. M. Wonham, “Supervisory control of a class of discrete event processes,” *SIAM J. Control Optim.*, vol. 25, no. 1, pp. 206–230, 1987.
[14] ——, “The control of discrete event systems,” Proc. of the IEEE, vol. 77, no. 1, pp. 81–98, 1989.
[15] K. R. Rohloff, “Sensor failure tolerant supervisory control,” in Proc. 44th IEEE Conf. Decision Control Eur. Control Conf. (CDC-ECC). IEEE, 2005, pp. 3493–3498.
[16] A. M. Sánchez and F. J. Montoya, “Safe supervisory control under observability failure,” Discrete Event Dyn. Syst.: Theory Applicat., vol. 16, no. 4, pp. 493–525, 2006.
[17] L. K. Carvalho, Y.-C. Wu, R. Kwong, and S. LaFortune, “Detection and prevention of actuator enablement attacks in supervisory control systems,” in Proc. 13th International Workshop on Discrete Event Systems (WODES). IEEE, 2016, pp. 298–305.
[18] Q. Wen, R. Kumar, J. Huang, and H. Liu, “A framework for fault-tolerant control of discrete event systems,” IEEE Trans. Autom. Control, vol. 53, no. 8, pp. 1839–1849, 2008.
[19] A. Paoli, M. Sartini, and S. LaFortune, “Active fault tolerant control of discrete event systems using online diagnostics,” Automatica, vol. 47, no. 4, pp. 639–649, 2011.
[20] S. Shu and F. Lin, “Fault-tolerant control for safety of discrete-event systems,” IEEE Trans. Autom. Sci. Eng., vol. 11, no. 1, pp. 78–89, 2014.
[21] S. Takai and T. Ushio, “Reliable decentralized supervisory control of discrete event systems,” IEEE Trans. Syst., Man, Cybern., B, Cybern., vol. 30, no. 5, pp. 661–667, 2000.
[22] F. Liu and H. Lin, “Reliable supervisory control for general architecture of decentralized discrete event systems,” Automatica, vol. 46, no. 9, pp. 1510–1516, 2010.
[23] M. V. Iordache and P. J. Antsaklis, “Resilience to failures and reconfigurations in the supervision based on place invariants,” in Proc. 2004 Amer. Control Conf. (ACC), 2004, pp. 4477–4482.
[24] J. Li, M. C. Zhou, T. Guo, Y. Gan, and X. Dai, “Robust control reconfiguration of resource allocation systems with Petri nets and integer programming,” Automatica, vol. 50, no. 3, pp. 915–923, 2014.
[25] M. Karimadini and H. Lin, “Fault-tolerant cooperative tasking for multi-agent systems,” Int. J. Control, vol. 84, no. 12, pp. 2092–2107, 2011.
[26] D. Angluin, “Learning regular sets from queries and counterexamples,” Inform. and Computation, vol. 75, no. 2, pp. 87–106, 1987.
[27] K. T. Seow, M. T. Pham, C. Ma, and M. Yokoo, “Coordination planning: Applying control synthesis methods for a class of distributed agents,” IEEE Trans. Control Syst. Technol., vol. 17, no. 2, pp. 405–415, 2009.
[28] K. Cai and W. M. Wonham, “Supervisor localization: a top-down approach to distributed control of discrete-event systems,” IEEE Trans. Autom. Control, vol. 55, no. 3, pp. 605–618, 2010.
[29] J. Komenda, T. Masopust, and J. H. van Schuppen, “Coordination control of discrete-event systems revisited,” Discrete Event Dyn. Syst.: Theory Applicat., vol. 25, no. 1-2, pp. 55–94, 2015.
[30] C. S. Păsăreanu, D. Giannakopoulou, M. G. Bobaru, J. M. Cobleigh, and H. Barringer, “Learning to divide and conquer: Applying the L* algorithm to automate assume-guarantee reasoning,” Formal Methods Syst. Des., vol. 32, no. 3, pp. 175–205, 2008.
[31] R. Kumar and V. K. Garg, Modeling and Control of Logical Discrete Event Systems. Boston: Kluwer, 1995.
[32] Y. Willner and M. Heymann, “Supervisory control of concurrent discrete-event systems,” Int. J. Control, vol. 54, no. 5, pp. 1143–1169, 1991.
[33] L. Lin, A. Stefanescu, and R. Su, “On distributed and parameterized supervisor synthesis problems,” IEEE Trans. Autom. Control, vol. 61, no. 3, pp. 777–782, 2016.
[34] J. Dai, A. Benini, H. Lin, P. J. Antsaklis, M. J. Rutherford, and K. P. Valavanis, “Learning-based formal synthesis of cooperative multi-agent systems,” arXiv preprint arXiv:1705.10427, 2017.
[35] M. Heymann and F. Lin, “On-line control of partially observed discrete event systems,” Discrete Event Dyn. Syst.: Theory Applicat., vol. 4, no. 3, pp. 221–236, 2014.
[36] H. Darabi, M. A. Jafari, and A. L. Buczak, “A control switching theory for supervisory control of discrete event systems,” IEEE Trans. Robot. Autom., vol. 19, no. 1, pp. 131–137, 2003.