A Framework for Control-Reconfiguration Following Fault-Detection in Discrete Event Systems

Ratnesh Kumar ∗ Shigemasa Takai ∗∗

∗ Iowa State University, Ames, IA 50011-3060 USA (e-mail: rkumar@iastate.edu).
∗∗ Osaka University, Suita, Osaka 565-0871, Japan (e-mail: takai@eei.eng.osaka-u.ac.jp).

Abstract: We introduce a framework for control-reconfiguration in discrete event systems that performs fault-detection and then reconfigures the control to ensure that certain desired specifications are met. The controlled system has separate specifications prior to a fault versus after a fault (to allow for any changes in the specification owing to the occurrence of a fault). Prior to the occurrence of a fault, a certain controller is applied so as to ensure a given pre-fault specification. The same controller is continued to be used even after a fault and until its detection. Once a fault is detected, the control is reconfigured. All along the post-fault duration (prior to as well as after reconfiguration), the controlled system is required to satisfy a given post-fault specification. We mathematically formulate the above control problem and provide a condition for reconfigurability. We also establish the condition for the limiting case where control-reconfiguration is not required. Also, we introduce the notion of degree of disambiguability of faulty traces which when increased, increases the ability to meet the control requirements of pre- and post-fault specifications.

Keywords: Discrete event system, Control-reconfiguration, Fault-tolerance, Supervisory control.

1. INTRODUCTION

Fault-tolerance is an important issue in design of dependable systems. Following the occurrence of a fault, a fault-tolerant system can continue its proper operation, although the performance may be degraded owing to the occurrence of a fault. Fault-tolerance can additionally require fault-recovery, meaning the system should resume functionality, fully or partially, within a bounded time. A framework for fault-tolerant control of Discrete Event Systems (DESs) (called plants) involving pre-fault versus post-fault specifications and fault-recovery was presented in Wen et al. (2008b).

One key aspect of fault-tolerance is fault-detection and control-reconfiguration, and this paper presents a framework for studying the same. Control-reconfiguration is performed following fault-detection, and until then a nominal controller is applied. The goal is to ensure that the controlled system satisfies a given pre-fault specification prior to a fault, and a given post-fault specification following a fault. Also, while a control-configuration occurs only after a fault is detected (which can be later than when a fault occurs), the controlled system continues to meet the respective specifications of the pre-fault and post-fault conditions. We mathematically formulate this problem and provide a necessary and sufficient condition for control-reconfigurability.

There has been some prior work on fault-tolerant control of DESs (see for example (Jensen, 2003)). Some involved controller switching upon the occurrence of a fault as in Darabi et al. (2003), or synthesizing a controller under partial observation as in Rohloff (2005). The resulting controlled system can tolerate some faults, but the system performance after faults will remain degraded since the notion of recovery from faults was not incorporated. Case studies involving synthesis of fault-tolerant supervisors can also be found in Cho and Lim (1996), Cho and Lim (1998), Zhou and Dicesare (1989). Design of certain coordination protocols for automated highway systems to achieve fault-tolerance under vehicle failures is reported in Lygeros et al. (2000), Godbole et al. (2000). Takai and Ushio (2000) considered the problem of reliable decentralized supervisory control, where they studied fault-tolerance with respect to the failures of the supervisors. Fault-tolerance in Petri nets is considered in Iordache and Antsaklis (2004), where liveness enforcing strategies are designed to deal with failures using system reconfigurations. In Lafortune and Lin (1991), the authors considered a pair of specifications, representing the desired and the (more liberal) tolerable behaviors for pre-fault and post-fault conditions, respectively, and additionally required a bounded-delay fault-recovery. Results for such fault-tolerant control synthesis were presented in Wen et al. (2008a). There also exists a large body of literature for fault-tolerance in software.
as well as computing systems. However, we focus here on physical systems, which, unlike a software or computing system, cannot be “rolled-back”.

Note the work on control-reconfiguration following fault-detection presented in this paper is in contrast to the framework presented in Wen et al. (2008b) that studied fault-tolerant control without having to perform reconfiguration (and as a result no fault-detection was required). Further, besides the satisfaction of pre-fault and post-fault specifications, Wen et al. (2008b) also required a fault-recovery, meaning reaching a state (after the occurrence of a fault) that is equivalent to a nonfaulty state so as to achieve fault-recovery as a required part of fault-tolerance. We have not considered fault-recovery in the present work, but our results can be extended (as part of future work) to also encompass fault-recovery.

Aside from studying the control-reconfiguration problem, we also show that when control-reconfiguration is not an option, a stronger condition must be satisfied, thereby quantifying the benefits of control-reconfiguration. Further, we establish a connection between the degree to which the faulty traces can be disambiguated and the ability to design reconfigurable-control: Larger the set of disambiguatable faulty traces, weaker the condition needed to meet the control requirements. So it makes sense to design systems with higher degree of disambiguatability that allows early detection of larger number of faults, thereby enabling early reconfiguration for larger number of faulty behaviors.

2. NOTATION AND PRELIMINARIES

We consider a DES modeled by an automaton \( G = (Q, \Sigma, \delta, q_0, M_0) \), where \( Q \) is the set of states, \( \Sigma \) is the finite set of events, a partial function \( \delta : Q \times \Sigma \rightarrow Q \) is the transition function, \( q_0 \in Q \) is the initial state, and \( Q_m \subseteq Q \) is the set of marked states. Let \( \Sigma^* \) be the set of all finite traces of elements of \( \Sigma \), including the empty trace \( \varepsilon \). The state transition function \( \delta \) can be generalized to \( \delta : Q \times \Sigma^* \rightarrow Q \) in the natural way. The generated and marked languages of \( G \), denoted by \( L(G) \) and \( L_m(G) \), respectively, are defined as \( L(G) = \{ s \in \Sigma^* \mid |\delta(q_0, s)| \} \) and \( L_m(G) = \{ s \in L(G) \mid |\delta(q_0, s)| \} \).

Let \( L \subseteq \Sigma^* \) be a language. We denote the set of all prefixes of traces in \( L \) by \( \bar{L} \), i.e., \( \bar{L} = \{ s \in \Sigma^* \mid \exists t \in \Sigma^*: st \in L \} \). \( L \) is said to be (prefix-)closed if \( \bar{L} = L \). We also denote \( \{0\} \) by \( \{0\} \) for each \( t \in \Sigma^* \). The set \( \bar{L} \) is the set of all extensions that can be executed in \( L \) after \( t \) has been executed. For any two languages \( L_1, L_2 \subseteq \Sigma^* \), their concatenation \( L_1 L_2 \subseteq \Sigma^* \) is defined as \( L_1 L_2 = \{ s \in \Sigma^* \mid s \in L_1, t \in L_2 \} \).

For supervisory control purposes (Ramadge and Wonham, 1987), the event set \( \Sigma \) is partitioned into two disjoint subsets \( \Sigma_c \) and \( \Sigma_u \) of controllable and uncontrollable events, respectively. We assume that a supervisor observes an event through the observation mask \( M : \Sigma \rightarrow \Delta \cup \{\varepsilon\} \) where \( \Delta \) is the set of observed symbols (Cieslak et al., 1988). The mask \( M \) is extended to \( M : \Sigma^* \rightarrow \Delta^* \) in the natural way. The inverse mask \( M^{-1} : \Delta^* \rightarrow 2^{\Sigma^*} \) is defined as \( M^{-1}(t) = \{ s \in \Sigma^* \mid M(s) = t \} \) for any \( t \in \Delta^* \).

Formally, a supervisor \( S \) is defined as \( S : \Delta^* \rightarrow 2^{\Sigma^e} \), where \( S(t) \subseteq \Sigma_c \) is the set of controllable events which are disabled after \( t \in \Delta^* \) is observed by \( S \). The generated language of the plant \( S/G \) supervised by \( S \) is defined inductively as follows:

- \( \varepsilon \in L(S/G) \),
- \( \forall s \in L(S/G), \forall \sigma \in \Sigma \) \( s\sigma \in L(S/G) \Leftrightarrow [s\sigma \in L(G) \land \sigma \notin S(M(s))] \).

A language \( L \subseteq L(G) \) is said to be

- controllable with respect to \( L(G) \) (Ramadge and Wonham, 1987) if \( TL\Sigma_u \cap L(G) \subseteq T \), and
- observable with respect to \( L(G) \) (Lin and Wonham, 1988; Cieslak et al., 1988) if, for any \( s, s' \in T \) and any \( \sigma \in \Sigma_c \),

\[
[M(s) = M(s') \land s\sigma \in T \land s'\sigma \in L(G)] \Rightarrow s'\sigma \in T.
\]

There always exists the infimal closed controllable and observable superlanguage of \( L \) with respect to \( L(G) \), denoted by \( \inf CO(L(G)) \) (Lin and Wonham, 1988).

For a nonempty closed language \( K \subseteq L(G) \), there exists a supervisor \( S : \Delta^* \rightarrow 2^{\Sigma_c} \) such that \( S(L(G) = K \) if and only if \( K \) is controllable and observable with respect to \( L(G) \) (Lin and Wonham, 1988; Cieslak et al., 1988).

3. CONTROL-RECONFIGURATION PROBLEM

In this section, we present a framework for control-reconfiguration following the detection of a fault. The overall plant model consists of a submodel specifying its nonfaulty part (i.e., the part prior to the occurrence of any fault). A model \( G = (Q, \Sigma, \delta, q_0) \) represents the overall behaviors of a plant consisting of behaviors prior to and subsequent to faults. Without loss of generality, we assume that the nonfaulty part of \( G \) is modeled as a subautomaton \( G^N = (Q^N, \Sigma, \delta^N, q_0) \) of \( G \) (meaning \( Q^N \subseteq Q \) and, for any \( q \in Q^N \) and \( \sigma \in \Sigma \) for which \( \delta^N(q, \sigma) \) is defined, \( \delta(q, \sigma) \) is also defined and \( \delta(q, \sigma) = \delta^N(q, \sigma) \)). Our framework allows the nonfaulty plant to have its own own control specification (a “pre-fault” specification), while the overall plant can have a separate post-fault control specification (since the occurrence of a fault may require an alteration in the specification for the controlled system behaviors). Let \( K^N \subseteq L(G^N) \) and \( K \subseteq L(G) \) be nonempty closed language specifications for pre-fault and post-fault conditions, respectively.

Example 1. As an example, consider the models \( G \) of the overall plant and \( G^N \) of the nonfaulty subplant shown in Fig. 1. Here the event set \( \Sigma = \{a, b, c, d, f\} \) and \( f \) denotes a faulty event. The models for pre- and post-fault specifications \( K^N \subseteq L(G^N) \) and \( K \subseteq L(G) \) are shown in Fig. 2 (a) and (b), respectively. \( K^N \) requires that in the pre-fault condition, \( b \) should never be allowed, and \( c \) should be blocked from occurring after the occurrence of \( d \), whereas \( K \) requires that in the post-fault condition, \( b \) should be blocked from occurring only after the occurrence of \( d \) and \( c \) should never be blocked.

The goal for control-reconfiguration is to design two controllers, one for prior to the detection of fault, and another for afterward such that the controlled system under their control satisfies the pre-fault specification prior to the
occurrence of fault and post-fault specification afterward. (Note the specification switches as soon as a fault occurs, but the control switches only after the detection of that fault.) For the reason that control cannot be reconfigured until after the detection of a fault, we characterize those faulty behaviors of post-fault specification for which ambiguity about the faults has not been resolved yet.

**Definition 2.** Given models $G^N$ and $G$ of nonfaulty subplant and overall plant, pre-fault and post-fault specifications $K^N \subseteq L(G^N)$ and $K \subseteq L(G)$, respectively, and an observation mask $M : \Sigma^* \rightarrow \Delta^*$, the set of ambiguous faulty traces in the post-fault specification, denoted by $K^\triangledown$, is given by:

$$K^\triangledown := \{s \in K - L(G^N) \mid M^{-1}M(s) \cap L(G^N) \cap K \neq \emptyset\}.$$

$K^\triangledown$ consists of faulty traces of $K$ that are indistinguishable from nonfaulty traces of $K$. Aside from the traces in $K^\triangledown$, the control cannot be reconfigured for also the traces belonging to the pre-fault specification. Thus the set of all traces for which control shall not be reconfigured is given by,

$$K^\triangledow := K^\triangledown \cup K^N.$$

Then the control-reconfiguration problem is to synthesize two supervisors $S^N : \Delta^* \rightarrow 2^{\Sigma_c}$ and $S : \Delta^* \rightarrow 2^{\Sigma_c}$ which satisfy

1. $L(S^N/G^N) = K^N$
2. $K^\triangledow \subseteq L(S^N/G) \subseteq K$, and
3. $L(S/G) = K$.

(1) states that the nonfaulty plant should satisfy the pre-fault specification under the control of supervisor prior to fault-detection; the second inequality of (2) and (3) together state that the overall plant should satisfy the post-fault specification prior to as well as after the control-reconfiguration (under the supervisors $S^N$ and $S$, respectively). (2) additionally states, through its first inequality, that once a fault happens (so the active plant model is no longer $G^N$ but $G$), the controlled system under $S^N$ (i.e., prior to control-reconfiguration) should include all traces for which control-reconfiguration is not allowed, and yet the controlled system continues to obey the post-fault specification (the second inequality of (2)).

4. **EXISTENCE OF SUPERVISORS**

In this section, we present a necessary and sufficient condition for the existence of supervisors which solve the proposed control-reconfiguration problem.

The following lemma is obvious and relates the behaviors of a plant and its subplant under the control of a common supervisor:

**Lemma 3.** Let $S : \Delta^* \rightarrow 2^{\Sigma_c}$ be any supervisor. For any plants $G_1$ and $G_2$ with $L(G_1) \subseteq L(G_2)$,

$$L(S/G_1) = L(S/G_2) \cap L(G_1).$$

Next, the following theorem is obtained that provides a desired necessary and sufficient condition.

**Theorem 4.** Given models $G^N$ and $G$ of nonfaulty subplant and overall plant, pre-fault and post-fault specifications $K^N \subseteq L(G^N)$ and $K \subseteq L(G)$, respectively, and an observation mask $M : \Sigma^* \rightarrow \Delta^*$, there exist supervisors $S^N : \Delta^* \rightarrow 2^{\Sigma_c}$ and $S : \Delta^* \rightarrow 2^{\Sigma_c}$ such that $L(S^N/G^N) = K^N$, $K^\triangledow \subseteq L(S^N/G) \subseteq K$, and $L(S/G) = K$ if and only if

- $K$ is controllable and observable with respect to $L(G)$,
- $\inf CCO_L(G)(K^\triangledow) \subseteq K$,
- $\inf CCO_L(G)(K^\triangledow) \cap L(G^N) = K^N$.

**Proof.** ($\Leftarrow$) Since $K$ is controllable and observable with respect to $L(G)$, there exists a supervisor $S : \Delta^* \rightarrow 2^{\Sigma_c}$ such that $L(S/G) = K$. Also, since $\inf CCO_L(G)(K^\triangledow)$ is controllable and observable with respect to $L(G)$, there exists a supervisor $S^N : \Delta^* \rightarrow 2^{\Sigma_c}$ such that $L(S^N/G^N) = \inf CCO_L(G)(K^\triangledow)$. Then, we have

$$K^\triangledow \subseteq L(S/G) = \inf CCO_L(G)(K^\triangledow) \subseteq K.$$  

Also, by Lemma 3, we have

$$L(S^N/G^N) = L(S^N/G) \cap L(G^N) = \inf CCO_L(G)(K^\triangledow) \cap L(G^N) = K^N.$$

($\Rightarrow$) Since there exist supervisors $S^N : \Delta^* \rightarrow 2^{\Sigma_c}$ and $S : \Delta^* \rightarrow 2^{\Sigma_c}$ such that $L(S/G) = K$ and $K^\triangledow \subseteq L(S^N/G) \subseteq K$, $K$ is controllable and observable with respect to $L(G)$ and $L(S^N/G)$ is a closed controllable and observable language with respect to $L(G)$ such that $K^\triangledow \subseteq L(S^N/G) \subseteq K$. It follows that

$$\inf CCO_L(G)(K^\triangledow) \subseteq L(S^N/G) \subseteq K.$$

Further, we have

![Fig. 1. Plant G and its nonfaulty part G^N.](image1)

![Fig. 2. Generators for pre-fault specification K^N and post-fault specification K.](image2)
We assume that the pre-fault specification is not violated by the post-fault specification uncontrollably or unobservable with respect to the overall plant.

Remark 5. By the proof of Theorem 4, if the proposed control-reconfiguration problem is solvable, supervisors $S$ and $S^N$ with $L(S/G) = K$ and $L(S^N/G) = \inf CCO_{L(G)}(K^o)$ solve it. It thus follows from the 2nd bullet of Theorem 4 that $L(S^N/G) \subseteq L(S/G)$, which is to be expected since the supervisor $S$ for the overall plant cannot omit any behaviors already executed under the supervisor $S^N$ for the nonfaulty plant; $S$ can only add to those behaviors.

Remark 6. Apart from the controllability and observability of post-fault specification with respect to the overall plant (first bullet), Theorem 2 requires that the execution of traces in $K^o$, prior to control-reconfiguration, will not violate the post-fault specification uncontrollably or unobservable (second bullet), and also such executions in the nonfaulty part (i.e., those included in $L(G^N)$) conform to the pre-fault specification (3rd bullet).

Example 7. We consider the plant $G$ and its nonfaulty part $G^N$ shown in Fig. 1. Let $\Sigma_c = \{a, b, c\}$, and $\Sigma_{uc} = \{d, f\}$, and

$$M(\sigma) = \begin{cases} \sigma, & \text{if} \ \sigma \in \{a, b, c\} \\ \varepsilon, & \text{otherwise}. \end{cases}$$

We assume that the pre-fault specification $K^N$ and the post-fault specification $K$ are generated by automata shown in Fig. 2 (a) and (b), respectively. We can verify that $K$ is controllable and observable with respect to $L(G)$.

In order to verify the 2nd and 3rd bullets of Theorem 2, we need to compute $\inf CCO_{L(G)}(K^o)$. We have $K^V = \{af, ef\}$, and $\inf CCO_{L(G)}(K^o)$ is generated by an automaton shown in Fig. 3. Then we can verify that $\inf CCO_{L(G)}(K^o) \subseteq K$ and $\inf CCO_{L(G)}(K^o) \cap L(G^N) = K^N$ hold. Finally, the generator for $\inf CCO_{L(G)}(K^o)$ of Fig. 3 serves as a supervisor $S^N$ that acts prior to the detection of a fault, whereas the generator for $K$ of Fig. 2(b) serves as a supervisor $S$ that acts after the detection of a fault. Note as an example of control-reconfiguration, $S^N$ disables the event $b$ following the observation of $a$ or $c$, whereas $S$ enables $b$ following those same observations.

4.1 When is Control-Reconfiguration not Needed

Theorem 4 provides a necessary and sufficient condition for control-reconfigurability. A natural question arises as to what additional properties must hold so that control-reconfiguration is not required, i.e., $S^N$ and $S$ can be taken to be the same supervisor. The answer comes from examining the conditions of Theorem 4. It can be seen that the first two bullets in Theorem 4 arise naturally from the requirements of the control-reconfiguration problem. However, the last bullet is revealing, which can equivalently be written in the form of the following inequality:

$$\inf CCO_{L(G)}(K^o) \cap L(G^N) \subseteq K^N,$$

as the reverse inequality trivially holds. Then it is evident that larger the language $K^o$, the stronger the requirement imposed by the above inequality becomes. Recall that $K^o$ captures all traces for which a control-reconfiguration cannot be allowed, and in the worst case $K^o$ can be such that $\inf CCO_{L(G)}(K^o)$ equals all of $K$, and in that case a control-reconfiguration will obviously be not possible. Note in this case (i.e., when $\inf CCO_{L(G)}(K^o) = K$), the second bullet of Theorem 4 trivially holds, whereas the third bullet becomes $K \cap L(G^N) = K^N$. Accordingly the following theorem characterizes the condition under which control-reconfiguration is not required.

Theorem 8. Given models $G^N$ and $G$ of nonfaulty subplant and overall plant, pre-fault and post-fault specifications $K^N \subseteq L(G^N)$ and $K \subseteq L(G)$, respectively, and an observation mask $M : \Sigma^* \rightarrow \Sigma^*$, there exist supervisors $S^N : \Delta^* \rightarrow 2^{2\Sigma}$ and $S : \Delta^* \rightarrow 2^{2\Sigma}$ such that $S^N = S$, $L(S^N/G^N) = K^N$, $K^o \subseteq L(S^N/G) \subseteq K$, and $L(S/G) = K$ if and only if $K$ is controllable and observable with respect to $L(G)$ and $K \cap L(G^N) = K^N$.

Proof. ($\Rightarrow$) Since $K$ is controllable and observable with respect to $L(G)$, there exists a supervisor $S : \Delta^* \rightarrow 2^{2\Sigma}$ such that $L(S/G) = K$. Let $S^N = S$, then by invoking Lemma 3 we can obtain:

$$L(S^N/G^N) = L(S/G^N) = L(S/G) \cap L(G^N) = K \cap L(G^N) = K^N.$$ 

Also, by the definition of $K^o$, we have

$$K^o \subseteq K \cup K^N$$

$$= K \cup (K \cap L(G^N))$$

$$= K$$

$$= L(S/G) = L(S^N/G^N).$$

($\Leftarrow$) Since $L(S/G) = K$, $K$ is controllable and observable with respect to $L(G)$. Further, since $L(S^N/G^N) = K^N$, where $S^N = S$, we have $L(S/G^N) = K^N$. Then, since $L(S/G^N) = L(S/G) \cap L(G^N)$ by Lemma 3, it follows that $K^N = K \cap L(G^N)$.

It can be concluded from Theorem 8 that if the second and the third bullets of Theorem 4 are replaced with the stronger requirement that $K \cap L(G^N) = K^N$, then control-reconfiguration is not needed.
4.2 Benefits of Reconfiguration & Disambiguability of Faulty Traces

Note the condition $K \cap L(G^N) = K^N$ of Theorem 8 implies that $K^N \subseteq K$ and so it follows that $K^\circ \subseteq K^N \cup (K - L(G^N)) \subseteq K$, and then under the controllability and observability of $K$, it also holds that $\text{inf } CCO_{L(G)}(K^\circ) \subseteq K$, i.e., the second bullet of Theorem 4 holds. It follows the conditions of controllability and observability of $K$ together with

$$\text{inf } CCO_{L(G)}(K^\circ) \subseteq K$$

hold regardless of whether a control-reconfiguration is required or not. The extra condition, needed for control-reconfiguration, of

$$\text{inf } CCO_{L(G)}(K^\circ) \cap L(G^N) = K^N$$

must be replaced with the stronger condition

$$K \cap L(G^N) = K^N$$

to ensure no control-reconfiguration.

Note since (1) must hold, the left hand side of (2) is smaller than the left hand side of (3), and so clearly (2) is easier to satisfy than (3). Thus the flexibility of having control-reconfiguration makes it easier to meet the control requirements (of pre-fault and post-fault specifications).

Furthermore, smaller the language $K^\circ$, the weaker the requirement imposed by (2). The language $K^\circ$ is smaller when more faulty traces of the post-fault specification are disambiguatable. In fact, the smallest value $K^\circ$ can take is when every fault is immediately detected so that $K^N = \emptyset$. Thus we can observe that an increase in the degree to which faults can be disambiguated enhances the ability to fulfill the control requirements (of pre-fault and post-fault specifications). We define the degree of disambiguability of faulty traces as its unambiguous portion:

**Definition 9.** Given models $G^N$ and $G$ of nonfaulty subplant and overall plant, a post-fault specification $K \subseteq L(G)$, and an observation mask $M : \Sigma^* \rightarrow \Delta^*$, the degree of disambiguability of the faulty traces in the post-fault specification $K - L(G^N)$ is defined by,

$$D(K - L(G^N)) := \{ s \in K - L(G^N) \mid M^{-1}(s) \cap K \cap L(G^N) = \emptyset \}.$$  

We claim that higher the degree of disambiguability, greater is the ability to design a reconfigurable-control.

**Theorem 10.** Given models $G^N$ and $G$ of nonfaulty subplant and overall plant, pre-fault and post-fault specifications $K^N \subseteq L(G^N)$ and $K \subseteq L(G)$, respectively, and an observation mask $M : \Sigma^* \rightarrow \Delta^*$, higher the degree of disambiguability $D(K - L(G^N))$ smaller the language $K^\circ$.

**Proof.** It follows from the definitions of $K^\circ$ and $D(K - L(G^N))$ that $K^\circ \cup D(K - L(G^N)) = K \cup K^N$ and $K^\circ \cap D(K - L(G^N)) = \emptyset$. Since the languages $K^\circ$ and $D(K - L(G^N))$ have the constant union $K \cup K^N$ and the empty intersection, when the degree of disambiguability $D(K - L(G^N))$ is higher, the set of traces $K^\circ$ for which control-reconfiguration is not allowable becomes smaller.

5. VERIFICATION OF CONTROL-RECONFIGURABILITY

In this section, we study how to verify the necessary and sufficient conditions provided in Theorem 4 for the existence of supervisors which solve the proposed control-reconfiguration problem.

The first bullet is the standard controllability and observability conditions which can be polynomially verified. The second bullet is equivalent to $K^\circ \subseteq K$ under controllability and observability of $K$. To check the second and third bullets, we need to compute $K^\circ$ and $\text{inf } CCO_{L(G)}(K^\circ)$. We next discuss the computation of $K^\circ$ whereas the computation of $\text{inf } CCO_{L(G)}(K^\circ)$ is standard from the literature.

Let $R^N = (X^N, \Sigma, \lambda^N, x_0^N, X^N)$ and $R^* = (X, \Sigma, \lambda, x_0, X)$ be acceptors for $K^N$ and $K$, respectively. i.e., $L(R^N) = L_m(R^N) = K^N$ and $L(R) = L_m(R) = K$. We augment the nonfaulty subplant model $G^N$ by adding a dump state $q_N^0 \not\in Q^N$. Formally, the augmented automaton $G_N$ is defined as $G_N = (\tilde{Q}^N, \Sigma, \delta_N, q_0, Q^N)$, where $\tilde{Q}^N = Q^N \cup \{ q_N^0 \}$, and the state transition function $\delta_N : \tilde{Q}^N \times \Sigma \rightarrow Q^N$ is defined as

$$\delta_N(q_N^0, \sigma) = \{ \delta_N(q_N^0, \sigma), \text{if } q_N^0 \in Q^N \text{ and } \delta(q_N^0, \sigma) \text{ is defined} \}$$

$$\text{otherwise}.$$  

Then we have $L(G_N) = \Sigma^*$. Further, for any $s \in \Sigma^*$, $s \not\in L(G^N)$ if and only if $\delta_N(q_0, s) = q_N^0$, $R^N$ and $R$ are augmented in the similar way and the resulting augmented automata are denoted by $\tilde{R}^N = (\tilde{X}^N, \Sigma, \tilde{\lambda}^N, x_0^N, \tilde{X}^N)$ and $\tilde{R} = (\tilde{X}, \Sigma, \tilde{\lambda}, x_0, \tilde{X})$, respectively, where $x_d^N \not\in X^N$ and $x_d \not\in X$ are dump states of $\tilde{R}^N$ and $\tilde{R}$, respectively. To construct an acceptor $R_0^N$ for $K^N$, we define a automaton $T = (Z, \Sigma^T, \gamma_0, Z_m)$ as follows:

- $Z = (Q^N \times \tilde{X}^N \times \tilde{X}) \times (Q^N \times \tilde{X})$.
- $Z_m = \{ (q_1^N, \tilde{x}_1^N, x_1, q_2^N, \tilde{x}_2) \in Z \mid [\tilde{x}_1^N \neq x_1^N] \lor [q_1^N = q_2^N \land \tilde{x}_1 \neq x_d \land q_1^N \neq q_2^N \land \tilde{x}_2 \neq x_d] \}.$
- $z_0 = (q_0, x_0^N, x_0, q_0)$. 
- $\Sigma^T = (\Sigma \cup \{ \varepsilon \}) \times (\Sigma \cup \{ \varepsilon \}) \setminus \{ (\varepsilon, \varepsilon) \}$.
- $\gamma : Z \times \Sigma^T \rightarrow Z$ is defined as follows: For each $z = (q_1^N, \tilde{x}_1^N, x_1, q_2^N, \tilde{x}_2)$ and $\sigma_T = (\sigma_1, \sigma_2) \in \Sigma^T$,

$$\gamma(z, \sigma_T)$$

is defined if and only if $M(\sigma_1) = M(\sigma_2)$.

If $\gamma(z, \sigma_T)$ is defined, then

$$\gamma(z, \sigma_T) = (q_1^N, \tilde{x}_1^N, x_1, q_2^N, \tilde{x}_2).$$

where

$$\tilde{q}_i^N = \{ \delta_N(q_i^N, \sigma), \text{if } \sigma \neq \varepsilon \}$$

$$\tilde{x}_i^N = \{ \tilde{\lambda}(\tilde{x}_i, \sigma), \text{if } \sigma \neq \varepsilon \}.$$
Then, we remove the second label from each transition of $T$, and trim the automaton to obtain $R^c$ with $L_m(R^c) = K^c$. It follows that $K^c$ is polynomially computable with complexity $O(Q_N^2 \cdot |X_N^c| \cdot |X|^2)$. For computing inf $CCO_{L(G)}(K^c)$, we can use the existing results whose complexity is known to be exponential (see Tsitsiklis (1989)).

Whether or not the general existence condition of Theorem 4 can be polynomially verified remains open at this point. In practice however, the special case of Theorem 8 holds, namely, the post-fault specification, when restricted to the nonfaulty plant, equals the pre-fault specification, and in this case no control reconfiguration is required—see Theorem 8. Also in this case, the existence condition as specified in Theorem 8 is polynomially verifiable: The complexities of verifying controllability and observability of $K$ with respect to $L(G)$ are $O(|Q| \cdot |X|)$ and $O(|Q| \cdot |X|^2)$, respectively, whereas the equality $K \cap L(G^N) = K^N$ can be verified in $O(Q_N^2 \cdot |X| \cdot |X|^N)$.

6. CONCLUSIONS

Systems are subject to faults owing to wear, tear, and malfunction. It is important then to provide measures for tolerating faults, and one mechanism for achieving this is to detect the occurrence of a fault and then reconfigure the control. An important question is what happens during the period where a fault has occurred but not yet detected. We provided an answer to this question by first presenting a formulation of the control-reconfiguration problem, and next providing a condition for control-reconfigurability. An important feature of the formulation is that it allows the flexibility of having separate pre-fault and post-fault specifications. Also, the formulation respects the restriction that a control can only be reconfigured after any ambiguity about the occurrence of a fault is fully resolved, and so during the post-fault condition and prior to its detection, the same control is applied as during the pre-fault condition. Also the formulation further ensures that the respective specifications of pre-fault and post-fault conditions are enforced through control before and after the control-reconfiguration.

We also presented the additional condition, which when satisfied, ensures that no control-reconfiguration is needed, i.e., a single controller can be designed to meet the pre-fault as well as the post-fault specifications. In other words, the advantage of having the flexibility of control-reconfiguration is that the control requirements of pre- and post-fault specifications can be satisfied under weaker conditions. Further, we introduced the notion of degree of disambiguability of faulty traces and showed that the existence condition becomes easier to satisfy as the degree of disambiguability of faulty traces is boosted. Thus there is benefit to having a higher degree of disambiguability that allows an early detection of faults for a larger number of faulty traces.

REFERENCES


