A Framework for Fault-Tolerant Control of Discrete Event Systems

Qin Wen, Student Member, IEEE, Ratnesh Kumar, Fellow, IEEE, Jing Huang, Student Member, IEEE, and Haifeng Liu, Member, IEEE
Department of Electrical and Computer Engineering
Iowa State University, Ames, Iowa 50011

Abstract—We introduce a framework for fault-tolerant supervisory control of discrete-event systems. Given a plant, possessing both faulty and nonfaulty behavior, and a submodel for just the nonfaulty part, the goal of fault-tolerant supervisory control is to enforce a certain specification for the nonfaulty plant and another (perhaps more liberal) specification for the overall plant, and further to ensure that the plant recovers from any fault within a bounded delay so that following the recovery the system state is equivalent to a nonfaulty state (as if no fault ever happened). The specification for the overall plant is more liberal compared to the one for the nonfaulty part since a degraded performance may be allowed after a fault has occurred. We formulate this notion of fault-tolerant supervisory control and provide a necessary and sufficient condition for the existence of such a supervisor. The condition involves the usual notions of controllability, observability and relative-closure, together with the notion of stability. An example of a power system is provided to illustrate the framework. We also propose a weaker notion of fault-tolerance where following the recovery, the system state is simulated by some nonfaulty state, i.e. behaviors following the recovery are also the behaviors from some faulty state. Also, we formulate the corresponding notion of weakly fault-tolerant supervisory control and present a necessary and sufficient condition (involving the notion of language-stability) for the its existence. We also introduce the notion of nonuniformly-bounded fault-tolerance (and its weak version) where the delay-bound for recovery is not uniformly bounded over the set of faulty traces, and show that when the plant model has finitely many states, this more general notion of fault-tolerance coincides with the one in which the delay-bound for recovery is uniformly bounded.

Keywords: discrete event systems, fault-tolerance, supervisory control, stability, language convergence

I. INTRODUCTION

Discrete Event Systems (DESs) are systems with discrete states that evolve in response to events [17]. Examples include manufacturing systems, communication protocols, reactive software, and asynchronous hardware. A goal of supervisory control [25], [17] of such systems is to enforce a given specification by minimally restricting the behavior of a given system (called plant). The supervisory role is characterized by the fact that at any given plant state, the supervisor determines a set of controllable events to be enabled, so that the plant evolves over enabled events (including the uncontrollable events) without violating a given specification.

In this paper, we introduce a framework for fault-tolerant supervisory control of DESs. Given a plant \( G \), possessing both faulty and nonfaulty behavior, and a submodel \( G^N \) for the nonfaulty part, the goal of fault-tolerant supervisory control is to enforce a certain specification \( K^N \) for the nonfaulty plant and another (perhaps more liberal) specification \( K \supseteq K^N \) for the overall plant \( G \), and further to ensure that the plant recovers from any fault within a bounded delay, so that following the recovery the system state is equivalent to a nonfaulty state (as if no fault ever happened). A fault is modeled as an uncontrollable event, occurrence of which causes a transition from the nonfaulty part to the faulty part. The specifications \( K \) and \( K^N \) can be used to specify both the safety and the progress requirements. Since a degraded performance may be tolerable after the occurrence of a fault, the second specification is more liberal than the first one (and so it allows a larger set of traces).

In [20], authors considered a pair of specifications, representing the desired and the (more liberal) tolerable behavior for a plant \( G \) and proposed a general solution of their problem. In our setting, we also have two specifications: One is a desired behavior for the system without faults, and the other is a desired behavior for the system with faults. The control goal in our setting includes also the fault-tolerance: Other than meeting the respective specifications, the controller needs to ensure that a recovery takes following any fault within a bounded delay.

There has been some prior work on fault-tolerant control of DESs (see for example [16]). Some prior approaches involved controller switching upon the occurrence of a fault as in [10], or re-computation of a controller as in [26]. The resulting controlled system can tolerate some faults but the system performance after faults can 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 [8], [9], [30]. Design of certain coordination protocols for automated highway systems to achieve fault-tolerance under vehicle failures is reported in [23], [12]. Takai et al. considered the problem of reliable decentralized supervisory control [28], where they studied fault-tolerance with respect to the failures of the supervisors. Fault-tolerance in Petri Net is considered in [14], where liveness enforcing strategies are designed to deal with failures using system reconfigurations.
Here we consider the general problem of fault-tolerant supervisory control with fault recovery. A supervisor is used not only to enforce certain control specifications but also to ensure recovery following any fault. We formulate and study the above fault-tolerant control problem and provide a necessary and sufficient condition for the existence of such a supervisor. The condition involves the usual notions of controllability, observability and relative-closure, together with the notion of stability. The state-stability property is used to establish bounded delay recovery from a fault [7], [24].

As mentioned above, by recovery we imply returning, within bounded delay, to a state that is equivalent to a nonfaulty state. In some applications, a weaker form of recovery may suffice where the behaviors following the recovery are also the behaviors from some nonfaulty state. Thus following the recovery, the system satisfies those properties that are also satisfied by the behaviors starting from some nonfaulty state. We study this weaker notion of fault-tolerant control, and give a necessary and sufficient condition for the existence of a weakly fault-tolerant supervisor. In contrast to the notion of stability. The state-stability property is used to ensure recovery following any fault. We formulate and not only to enforce certain control specifications but also supervisory control with fault recovery. A supervisor is used to establish bounded delay recovery from a fault [7], [24].

The remainder of this paper is organized as follows. Section II gives the basic notation and preliminaries. Section III presents the concept of fault-tolerant supervisory control. Section IV introduces a condition for the existence of a fault-tolerant supervisor. Section V illustrated our framework by applying to a simplified power system. Section VI introduces the notion of weak fault-tolerance, and Section VII presents a condition for the existence of a weakly fault-tolerant supervisor, and Section VIII gives an example for weakly fault-tolerant supervisor. Section IX introduces and studies the notion of nonuniformlybounded fault-tolerance. Section X concludes the paper.

II. Notation and Preliminaries

A DES to be controlled, called plant, is modeled as an automaton, denoted by a five tuple \( G := (X, \Sigma, \alpha, x_0, X_m) \), where \( X \) denotes the set of states, \( \Sigma \) denotes the finite set of events, \( \alpha : X \times \Sigma \rightarrow X \) denotes the partial deterministic state transition function, \( x_0 \in X \) denotes the initial state, and \( X_m \subseteq X \) denotes the set of marked states. For \( x \in X \), we use \( \Sigma(x) \subseteq \Sigma \) to denote the set of events defined at \( x \), i.e., \( \Sigma(x) := \{ \sigma \in \Sigma \mid \alpha(x, \sigma) \text{ is defined} \} \). \( \Sigma^* \) is used to denote the set of all finite-length sequences of events, called traces, which includes the zero-length trace \( \epsilon \). The length of a trace \( s \), denoted as \( |s| \), is defined to be the number of events in the trace. A subset of \( \Sigma^* \) is called a language. The generated language of \( G \), denoted as \( L(G) \subseteq \Sigma^* \), contains all traces \( s \) for which \( \alpha(x_0, s) \) is defined. The marked language of \( G \), denoted as \( L_m(G) \), contains all generated traces that reach a marked state.

Given two automata \( G_1 := (X_1, \Sigma_1, \alpha_1, x_{01}, X_{m1}) \) and \( G_2 := (X_2, \Sigma_2, \alpha_2, x_{02}, X_{m2}) \), \( G_1 \) is said to be a subautomaton of \( G_2 \), denoted as \( G_1 \sqsubseteq G_2 \), if there exists an injective map \( h : X_1 \rightarrow X_2 \) such that \( \forall s \in L(G_1) : h(\alpha_1(x_{01}, s)) = \alpha_2(x_{02}, s) \).

For traces \( s \) and \( t \), we use \( s \leq t \) to denote that \( s \) is a prefix of \( t \) and \( s < t \) to denote that \( s \) is a proper prefix of \( t \). For a language \( K \subseteq \Sigma^* \), \( pr(K) \), called the prefix-closure of \( K \), denotes the set of all prefixes of \( K \), i.e., \( pr(K) = \{ s \in \Sigma^* \mid \exists t \in K : s \leq t \} \). It is clear that \( K \subseteq pr(K) \), and \( K \) is said to be prefix-closed if \( K = pr(K) \). We use \( K \setminus s \) to denote the set of traces that occur in the language \( K \) after the trace \( s \) has occurred, i.e., \( L\setminus s := \{ t \in \Sigma^* \mid sl \in L \} \). The language \( L_1 \setminus K_2 \), denoted as \( K_1 \setminus K_2 \), is defined as \( K_1 \setminus K_2 := \{ t \in \Sigma^* \mid \exists s \in K_2 \text{ such that } st \in K_1 \} \). For traces \( s \) and \( t \), we use \( s \sqsubseteq_G t \) to denote that the sets of traces that occur in the generated and the marked languages of \( G \) after \( s \) are contained in those after \( t \), i.e., \( L(G) \setminus s \subseteq L(G) \setminus t \) and \( L_m(G) \setminus s \subseteq L_m(G) \setminus t \). We write \( s \equiv_G t \) if \( s \sqsubseteq_G t \) and \( t \sqsubseteq_G s \) implies the equivalence of the behaviors following \( s \) and \( t \), whereas \( s \sqsubseteq_G t \) implies the behaviors following \( s \) and \( t \) are subsumed by the behaviors following \( t \). For control purposes, the event set of \( G \) is partitioned into the set of controllable events \( \Sigma_c \subseteq \Sigma \) and the set of uncontrollable events \( \Sigma_u \subseteq \Sigma \). A language \( K \) is said to be controllable (with respect to \( G \) and \( \Sigma_u \)) if \( pr(K) \cap \Sigma_u \subseteq pr(K) \). The events executed by the plant are filtered by an observation mask \( M : \Sigma \rightarrow \Delta \cup \{ \epsilon \} \) that maps the set of events to the set of “observed events” \( \Delta \). A language \( K \) is said to be observable (with respect to \( G \) and mask \( M \)) if \( \forall s, t \in pr(K) : \sigma \in \Sigma : M(s) = M(t), s\sigma \in pr(K), t\sigma \in L(G) \Rightarrow t\sigma \in pr(K) \). A language \( K \) is said to be relative-closed with respect to \( G \), if \( pr(K) \cap L_m(G) = K \cap L_m(G) \).

A supervisor is another automaton \( S := (Y, \Sigma, \beta, y_0, Y_m) \). The supervised plant is the synchronous composition of \( G \) and \( S \), denoted \( G||S := (X \times Y, \Sigma, \gamma, (x_0, y_0), X_m \times Y_m) \), where for \( (x, y) \in X \times Y \) and \( \sigma \in \Sigma \), \( \gamma((x, y), \sigma) \) is defined if and only if both \( \alpha(x, \sigma) \) and \( \beta(y, \sigma) \) are defined and in which case, \( \gamma((x, y), \sigma) = (\alpha(x, \sigma), \beta(y, \sigma)) \). It can be concluded that the generated and the marked languages of the supervised plant are \( L(G||S) = L(G) \cap L(S) \) and \( L_m(G||S) = L_m(G) \cap L_m(S) \), respectively.

A supervisor \( S \) is said to be (i) nonmarking if \( L_m(G||S) = L(G||S) \cap L_m(G) \), (ii) nonblocking if \( pr(L_m(G||S)) = \).
\(L(G||S), (iii) \Sigma_u\)-compatible if it does not disable any uncontrollable event (equivalently if \(L(G||S)\) is controllable), (iv) \(M\)-compatible if the controls following the indistinguishable traces are identical (equivalently if \(L(G||S)\) is observable), and (v) \((\Sigma, M)\)-compatible if it is both \(\Sigma_u\)-compatible and \(M\)-compatible (See for example [17]).

It is known that given a nonempty specification language \(K \subseteq L_m(G)\), there exists a \((\Sigma, M)\)-compatible, nonmarking and nonblocking supervisor if and only if \(K\) is relative-closed, controllable and observable [22].

For discrete event systems, there are two forms of stability: state-stability, as introduced in [7], [24], and language-stability, as introduced in [18], [29]. We first discuss the notion of state-stability.

Given \(\hat{X} \subseteq X, x \in X\) is \(\hat{X}\)-attractable in \(G\) if there exists a non-negative integer \(N\) such that for all traces \(t\) from \(x\) that are either deadlocking or have length greater than or equal to \(m\), \(t\) visits \(\hat{X}\). \(x \in X\) is controllably \(\hat{X}\)-attractable in \(G\) if there exists a supervisor \(S\) such that \(x\) is \(\hat{X}\)-attractable in \(G\|S\). We use \(\Omega_G(\hat{X})\), called the region of attraction of \(\hat{X}\), to denote the set of all \(\hat{X}\)-attractable states, and \(\hat{X}\) is called an attractor for the set \(\Omega_G(\hat{X})\). We use \(\Omega_G(\hat{X})\), called the region of controllable attraction of \(\hat{X}\), to denote the set of all controllably \(\hat{X}\)-attractable states, and \(\hat{X}\) is called a controllable attractor for the set \(\Omega_G^c(\hat{X})\). A state set \(\hat{X} \subseteq X\) is said to be controllable to \(\hat{X}\) if \(\hat{X} \subseteq \Omega_G^c(\hat{X})\) and controllably attractive to \(\hat{X}\) if \(\hat{X} \subseteq \Omega_G^c(\hat{X}) \subseteq \Omega_G(\hat{X})\). Clearly, \(\hat{X} \subseteq \Omega_G(\hat{X}) \subseteq \Omega_G^c(\hat{X})\).

A language \(L \subseteq \Sigma^*\) is said to be language-stable (\(\ell\)-stable) with respect to language \(K \subseteq \Sigma^*\), or converges to \(K\), if there exists \(m \in N\) such that for all \(s \in L\) with \(|s| \geq m\) or \(s\) deadlocks, exist \(s' \leq s\) and \(v \in K\) with \(|s'| \leq m\) and \(s = s'v\). In this case \(m\) is said to be the delay-bound of convergence. It follows from the definition that \(L\) is \(\ell\)-stable with respect to \(K\) if for every trace \(s \in L\) longer than \(m\) or is deadlocking, there exists a prefix of length at most \(m\) after which the corresponding suffix belongs to \(K\). Further, a language \(L \subseteq \Sigma^*\) is said to be language-stabilizable \((\ell\)-stabilizable\) with respect to \(K\), if there exists a supervisor \(S\) such that \(L(G||S)\) is \(\ell\)-stable with respect to \(K\).

### III. Fault-Tolerant Supervisory Control

In this section, we introduce a notion of fault-tolerant supervisory control. Consider a plant with model \(G = (X, \Sigma, \alpha, x_0, X_m)\) which represents the behavior prior to as well as subsequent to faults, i.e., the overall behavior. Let the nonfaulty part of the plant \(G\) be modeled as \(G^N = (X^N, \Sigma, \alpha^N, x_0, X_m^N)\). Without loss of generality, \(G^N \subseteq G\), i.e., \(G^N\) is a subautomaton of \(G\). The pair \((G, G^N)\) is said to be fault-tolerant if every post-fault behavior becomes equivalent to a nonfaulty behavior in a uniformly bounded delay. This property is captured as follows:

**Definition 1:** Given a plant \(G\) with its nonfaulty part \(G^N\), \((G, G^N)\) is said to be fault-tolerant if it exists \(m \in N\) such that for \(s \in L(G) - L(G^N)\), \(st \in L(G)\) with \(|t| \geq m\) or \(st\) deadlocks, there exist \(u \in L(G^N)\) and \(t' \leq t\) with \(|t'| \leq m\) and \(st' \equiv_G u\). In this case, \(m\) is called the delay-bound of fault-tolerance.

The plant represented as the pair \((G, G^N)\) is fault-tolerant if within a uniformly bounded delay of the occurrence of a fault, the plant state returns to a state equivalent to a nonfaulty state. Then the ensuing behavior is such that no fault ever happened. Therefore, after recovery, the system assumes full functionality. When the system model is minimal, i.e., possessing a minimal number of states, this means that following recovery, the system reaches a nonfaulty state. This, however, may not hold in general as shown in Figure 1, where \(G_{min}\) represents a minimal model of \(G\). (The dashed edges represent uncontrollable transitions.) Following a fault, \(G_{min}\) recovers to a nonfaulty state in one transition and so clearly it is fault-tolerant. This is not the case for the model \(G\) but, being behaviorally equivalent to \(G_{min}\), \(G\) is also fault-tolerant. (In Figure 1, state \(x\) is equivalent to nonfaulty state \(x\).) Our definition is behavior-based and captures this situation.

The following example illustrates a fault-tolerant plant model.

**Example 1:** Consider the plant \(G\) and its nonfaulty part \(G^N\) shown in Figure 2 that models a machine. Initially the machine is idle and nonfaulty. The start event \(a\) transitions the machine to working and nonfaulty state, and the stop event \(b\) brings it back to the initial state. In the working and nonfaulty state, an occurrence of the fault event \(f\) causes the machine to transition to the working and faulty state. An execution of \(b\) at this state causes the machine to move to the idle and faulty state from where the repair event \(c\) brings the machine back to the initial state. An execution of the repair event in the idle and nonfaulty state does not change the machine state. It can be verified that \(G\) is minimal.

When there is an exit from the nonfaulty part due to the execution of \(f\), a return within a bounded delay is not guaranteed since there exists a cycle between the two faulty states. It follows that \((G, G^N)\) is not fault-tolerant.
The above notion of fault-tolerance is a type of state-stability property. This is established in the following theorem.

**Theorem 1:** Consider a plant $G = (X, \Sigma, \delta, x_0, X_m)$ and its nonfaulty part $G^N = (X^N, \Sigma, \delta^N, x_0, X_m)$, and suppose the corresponding minimal plant and its nonfaulty part are $G_{min} = (X_{min}, \Sigma, \delta_{min}, x_{0, min}, X_{m, min})$ and $G^N_{min} = (X^N_{min}, \Sigma, \delta^N_{min}, x_{0, min}, X^N_{m, min})$ respectively. $(G, G^N)$ is fault-tolerant if and only if $X_{min}$ is attractable to $X^N_{min}$, i.e.,

\[ X_{min} \subseteq \Omega_{G_{min}}(X^N_{min}). \]

**Proof:** Since $G_{min}$ and $G^N_{min}$ are minimal models of $G$ and $G^N$, $L(G_{min}) = L(G)$, $L(G^N_{min}) = L(G^N)$, and since for any $s, t \in L(G) = L(G_{min})$, $s \equiv_G t \iff s \equiv_{G_{min}} t$. Therefore, $(G, G^N)$ is fault-tolerant if and only $(G_{min}, G^N_{min})$ is fault-tolerant.

Also note that $X_{min} \subseteq \Omega_{G_{min}}(X^N_{min})$ is equivalent to $X_{min} - X^N_{min} \subseteq \Omega_{G_{min}}(X^N_{min})$.

\[(\Rightarrow) \text{ Pick } s \in L(G_{min}) - L(G^N_{min}). \text{ Since } s \in L(G_{min}) - L(G^N_{min}), \text{ exists } x \in X_{min} - X^N_{min} \text{ such that } \delta_{min}(x, s) = x. \text{ Since } x \in X_{min} - X^N_{min} \subseteq \Omega_{G_{min}}(X^N_{min}), \text{ exists } m > 0 \text{ such that, for all } t, \text{ for which } \delta_{min}(x, t) \text{ is defined and either } |t| \geq m \text{ or } \delta_{min}(x, t) \text{ is deadlocking, exists } t' \leq t \text{ such that } \delta_{min}(x, t') \in X_{min}. \text{ It shows that } m \text{ is the desired delay bound for fault-tolerance. To see this, pick } t \text{ such that } st \in L(G_{min}) \text{ and either } |t| \geq m \text{ or } st \text{ is deadlocking. Then } \delta_{min}(x, t') \in X_{min} \text{ for some } t' \leq t. \text{ Let } u \in L(G^N_{min}) \text{ be such that } \delta_{min}(x_0, u) = \delta_{min}(x, t'). \text{ Then } u \text{ and } st' \text{ reach the same state in } G_{min}, \text{ so } u \equiv_{G_{min}} st'. \]

\[(\Leftarrow) \text{ Now assuming } (G_{min}, G^N_{min}) \text{ to be fault-tolerant, we establish that } X_{min} - X^N_{min} \subseteq \Omega_{G_{min}}(X^N_{min}). \text{ Pick } x \in X_{min} - X^N_{min}. \text{ Then there exists } s \in L(G_{min}) - L(G^N_{min}) \text{ such that } \delta_{min}(x_0, s) = x. \text{ From the fault-tolerance of } (G_{min}, G^N_{min}), \text{ there exists } m > 0 \text{ such that, for all } t \text{ with } st \in L(G_{min}) \text{ and either } |t| \geq m \text{ or } st \text{ is deadlocking, exists } u \in L(G^N_{min}) \text{ and } t' \leq t \text{ satisfying } u \equiv_{G_{min}} st'. \text{ From the minimality of } G_{min}, \text{ equivalence of } u \text{ and } st' \text{ implies they reach the same state. Since } u \in L(G^N_{min}), \delta_{min}(x_0, st') = \delta_{min}(x_0, u) \in X_{min}. \text{ It follows that, for each } t \text{ such that } \delta_{min}(x, t) \text{ is defined and } |t| \geq m \text{ or } \delta_{min}(x, t) \text{ is deadlocking, exist } t' \leq t \text{ such that } \delta_{min}(x, t') \in X_{min}. \text{ This implies } X_{min} - X^N_{min} \subseteq \Omega_{G_{min}}(X^N_{min}). \]

A given plant $(G, G^N)$ may not be intrinsically fault-tolerant but could be made so through the use of control. This motivates us to formulate the notion of a fault-tolerant supervisor, which exercises appropriate control actions so that the controlled plant $(G||S, G^N||S)$ is fault-tolerant. The control actions of a fault-tolerant supervisor ensure that following any fault, a recovery takes place within a bounded number of steps, i.e., the controlled plant state returns to a state from where the future behaviors are such that as if no fault ever happened.

**Definition 2:** Given a plant $G$ with its nonfaulty part $G^N$, a supervisor $S$ is said to be fault-tolerant if $(G||S, G^N||S)$ is fault-tolerant.

The following example illustrates the notion of fault-tolerance of a supervisor.

**Example 2:** Consider the example of Figure 2. The start event $\alpha$ is controllable and disabled by a supervisor $S$ at the idle and faulty state. The controlled systems $(G||S, G^N||S)$ is shown in Figure 3. It can be seen that from any faulty state a return to some nonfaulty state is guaranteed within at most two transitions, i.e., $(G||S, G^N||S)$ is fault-tolerant, or equivalently, $S$ is a fault-tolerant supervisor for $(G, G^N)$.

The following corollary follows from Theorem 1.

**Corollary 1:** Given a plant $G$ and its nonfaulty part $G^N$, and their corresponding minimal plant $G_{min}$ and $G^N_{min}$, exists a fault-tolerant supervisor $S$ if and only if $X_{min} \subseteq \Omega_{G_{min}}(X^N_{min})$.

**Proof:** $X_{min} \subseteq \Omega_{G_{min}}(X^N_{min})$ if and only if exists supervisor $S$ such that $X_{min} \subseteq \Omega_{G_{min}}(X^N_{min})$, or equivalently exists supervisor $S$ such that $(G_{min}, G^N_{min})$ is fault-tolerant. (The last equivalence follows from Theorem 1.)

**IV. EXISTENCE OF FAULT-TOLENTANT SUPERVISOR**

The previous section formulated the notion of a fault-tolerant supervisor as one that ensures recovery from a fault within a bounded number of steps. In general, a supervisor needs to enforce certain other control specifications. For example, in Figure 3, the plant possesses certain illegal and certain final states. The supervisor must also ensure that the illegal states are never visited while the final states are always reachable. To capture such control requirements, we use a pair of specification languages $K^N \subseteq L_m(G^N)$ and $K \subseteq L_m(G)$ satisfying $K^N \subseteq K$. Here $K^N$ represents the control specification for the nonfaulty plant, and a different control specification, namely $K$, is used for the overall plant. This specification is taken to be “more liberal” ($K \supseteq K^N$) since a downgraded performance may be tolerable after a fault has occurred.

Thus a fault-tolerant supervisory control problem is formulated as follows: Given a plant $G$, its nonfaulty part $G^N$, specifications $K$ and $K^N$ (with $K^N \subseteq K$), find a nonmarking, nonblocking, $(\Sigma_u, M)$-compatible and fault-tolerant supervisor $S$ such that $L_m(G^N||S) = K^N$ and $L_m(G||S) = K$.

A necessary and sufficient condition for this is provided in the following theorem.

**Theorem 2:** Given a plant $G = (X, \Sigma, \alpha, x_0, X_m)$ with nonfaulty part $G^N = (X^N, \Sigma, \alpha^N, x_0, X^N_m)$, specification $\emptyset \neq K \subseteq L_m(G)$ for $G$ and specification $\emptyset \neq K^N \subseteq L_m(G^N)$ for $G^N$ satisfying $K^N \subseteq K$, there exists a nonmarking, nonblocking (with respect to both $G^N$ and $G$), $(\Sigma_u, M)$-compatible and fault-tolerant supervisor $S$ such that

1) $L_m(G^N||S) = K^N$, $L(G^N||S) = pr(L_m(G^N||S))$, and
2) $L_m(G||S) = K$ and $L(G||S) = pr(L_m(G||S))$.

if and only if
1) $K$ is relative-closed, controllable and observable with respect to $G$.
2) In a minimal $R = (Q, \Sigma, \alpha, q_0, Q_m)$ and $R^N = (Q^N, \Sigma, \alpha^N, q_0, Q^N_m)$ with $R^N \subseteq R$, $L_m(R^N) = K^N$, and $L_m(R) = K$, it holds that $Q \subseteq \Omega_R(Q^N)$.
3) $K^N = K \cap L_m(G^N)$, and $pr(K^N) = pr(K) \cap L(G^N)$.

\textbf{Proof:} From [17], we know there exists a $(\Sigma_u, M)$-compatible, nonmarking and nonblocking supervisor $S$ such that $L_m(G||S) = K$ and $L(G||S) = pr(K)$ if and only if $K$ is relative-closed, controllable and observable with respect to $G$. $R$ and $R^N$ accept the same languages as $G||S$ and $G^N||S$, and they are minimal. From Theorem 1, we know $(G||S, G^N||S)$ is fault-tolerant if and only if in a minimal model $((G||S)_{min}, (G^N||S)_{min}) = (R, R^N)$ it holds that $Q \subseteq \Omega_R(Q^N)$.

So we only need to show that given $L_m(G||S) = K$ and $L(G||S) = pr(K)$, we have $L_m(G^N||S) = K^N$ and $L(G^N||S) = pr(K^N)$ if and only if $K$ and $K^N$ are constrained by $K^N = K \cap L_m(G^N)$ and $pr(K^N) = pr(K) \cap L(G^N)$. This follows from the following two series of equalities.

$$K^N = L_m(G^N||S)$$
$$= L_m(G^N) \cap L_m(S)$$
$$= L_m(G^N) \cap L_m(S) \cap L_m(G)$$
$$= L_m(G^N) \cap L_m(G||S)$$
$$= L_m(G^N) \cap K$$
and
$$pr(K^N) = L(G^N||S)$$
$$= L(G^N) \cap L(S)$$
$$= L(G^N) \cap L(S) \cap L(G)$$
$$= L(G^N) \cap L(G||S)$$
$$= L(G^N) \cap pr(K).$$

Remark 1: In Condition 1 of Theorem 2, the relative-closure property can be checked in $O(|G||R|)$ [17], the controllability property can be checked in $O(|G||R|)$, and observability property can be checked in $O(|G||R|^2)$. Condition 2 can be checked in $O(|Q|)$ [7]. Both parts of Condition 3 can be checked in $O(|G||R|)$. (The two parts of condition can be checked by checking in $G||R$ whether $X^N_m \times Q_m \subseteq X^N \times Q^N_m$ and $X^N \times R \subseteq X^N \times R^N$, respectively.) Thus the overall complexity of verifying the condition of Theorem 2 is $O(|G||R|^2)$.

In Theorem 2, we allowed $(K, K^N)$ to be an arbitrary pair of languages, and together they can capture both safety and liveness requirements. In some applications, the specification for the overall plant can simply be a safety specification, i.e., $K = pr(K)$. Theorem 2 can be specialized to address this situation resulting in the following corollary:

\textbf{Corollary 2:} Given a plant $G = (X, \Sigma, \alpha, x_0, X_m)$ with nonfaulty part $G^N = (X^N, \Sigma, \alpha^N, x_0, X^N_m)$, specification $\emptyset \neq K \subseteq L_m(G)$ for $G$ and specification $\emptyset \neq K^N \subseteq L_m(G^N)$ for $G^N$ satisfying $K^N \subseteq K$, there exists a nonmarking, nonblocking, $(\Sigma_u, M)$-compatible and fault-tolerant supervisor $S$ such that $K^N = K \cap L_m(G^N)$, $L(G^N||S) = pr(L_m(G^N||S))$, and $L(G||S) = K$.

if and only if
1) $K$ is prefix-closed, controllable and observable with respect to $G$.
2) In a minimal $R = (Q, \Sigma, \alpha, q_0, Q_m)$ and $R = (Q^N, \Sigma, \alpha^N, q_0, Q^N_m)$ with $R^N \subseteq R$, $L_m(R^N) = K^N$ and $L(R) = K$, it holds that $Q - Q^N \subseteq \Omega_R(Q^N)$.
3) $K^N = K \cap L_m(G^N)$, and
4) $pr(K^N) = K \cap L(G^N)$.

The proof of Corollary 2 is similar to that of Theorem 2 and is omitted for brevity.

\section{APPLICATION EXAMPLE}

We provide an application to illustrate our fault-tolerant control framework developed above by examining an abstracted model of a power system [2] shown in Figure 4. In this system, there are three generators ($G_1$, $G_2$ and $G_3$) and three loads (Load A, B and C), connected by nine buses (1 - 9).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{power_system.png}
\caption{A 9-bus power system}
\end{figure}

The states of a power system are typically categorized into six different classes depending on the stability, safety, performance, and security margin properties they possess (as shown in the table below). Stability of the system refers to the stability of the individual generators. Safety refers to the line powers and bus voltages being within an acceptable threshold. The system is said to have normal performance when all load demands are met, and otherwise the performance is said to be Degraded (as some loads may be shed and some generators may be switched-off). Security Margin refers to the margin by which system load may be increased without violating stability or safety.

<table>
<thead>
<tr>
<th>Category</th>
<th>Stable</th>
<th>Safe</th>
<th>Performance</th>
<th>Margin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal (N)</td>
<td>yes</td>
<td>yes</td>
<td>normal</td>
<td>large</td>
</tr>
<tr>
<td>Alert (A)</td>
<td>yes</td>
<td>yes</td>
<td>normal</td>
<td>small</td>
</tr>
<tr>
<td>Emergency (E)</td>
<td>yes</td>
<td>no</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>In-extremis (I)</td>
<td>no</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Recovery (R)</td>
<td>yes</td>
<td>yes</td>
<td>Degraded</td>
<td>-</td>
</tr>
<tr>
<td>Failed (F)</td>
<td>-</td>
<td>-</td>
<td>None</td>
<td>-</td>
</tr>
</tbody>
</table>

A normal state is one where system behavior is acceptable with respect to all four properties, and security margin is large enough that the occurrence of a single fault keeps the system...
behavior acceptable. An alert state is one where the system behavior is acceptable but security margin is small so that the occurrence of a single fault causes the system behavior to become unacceptable (i.e., one of the four properties may be violated). An emergency state is one where stability is not violated but safety is violated, whereas an in-extremis state is one where the stability is violated. A recovery state is one where safety and stability are not violated but the system performance is downgraded. Finally a failed state is one where the system is out-of-service.

Some features of the power system shown in Figure 4 are listed below:

1) The feasible uncontrollable events are the two line-faults:

\[
\begin{align*}
& f_1 \text{ Power line between buses 5 & 7 faulted} \\
& f_2 \text{ Power line between buses 5 & 4 faulted}
\end{align*}
\]

2) The feasible controllable events are:

- \( c \) Switch-on capacitor at bus 5
- \( p \) Switch-off capacitor at bus 5
- \( p_0 \) Switch-on power control at generators/loads prior to the critical time
- \( p_1 \) Switch-on power control at generators/loads after the critical time
- \( b \) Switch-off power control at generators/loads
- \( r \) Repair a faulted line (when system is stable)

3) States are abstracted to represent the values of the 4 binary state-variables: \( \{s_1, s_2, s_3, s_4\} \), the meaning of which is listed as follows:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Meaning</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_1 )</td>
<td>Line 5-9 disconnected?</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>( s_2 )</td>
<td>Line 5-4 disconnected?</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>( s_3 )</td>
<td>Capacitor control on?</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>( s_4 )</td>
<td>Power control on?</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

There is an additional special state, namely the “Failed” state denoted as “F”, in which the system is out-of-service.

4) No control is exercised that takes the system to a “worse” state (for example, the control action of “switching off the capacitor” is not allowed in the alert state 1010 as this will cause a transition to the emergency state 1000).

5) We assume that no fault occurs in a recovery state.

The abstracted model of the power system is shown in Figure 5. In Figure 5, there are 16 “non-failed” states, and depending on their stability, safety, performance, and security margin properties are classified into one of five categories: normal (0010), alert (0000, 1010, 0110), recovery (---1), emergency (1000, 0100), and in-extremis (11-0). (Note that “-” represents a 0 or a 1.) In Figure 5 all solid edges are controllable and all dashed edges are uncontrollable. The only uncontrollable events are the two line-fault events. All events including the line-fault events are observable. (When a line-fault occurs, the circuit breakers at the ends of the line open-up to disconnect that line and that information can be used to determine the occurrence of a line-fault.)

The normal operation corresponds to no faults \( (s_1 = s_2 = 0) \), capacitive control on \( (s_3 = 1) \), and power control off \( (s_4 = 0) \). In this state the system behaves acceptable and has large security margin. Switching the capacitive control off or the occurrence of a single fault causes the security margin to become smaller and the system transitions to one of the alert states 0000 (when capacitive control is switched off), or 1010 (when fault \( f_1 \) occurs), or 0110 (when fault \( f_2 \) occurs). In state 0000 no fault has occurred but the capacitive control is off. Due to this, the occurrence of either fault \( f_1 \) or \( f_2 \) causes the system to violate safety (voltage at bus 5 dipping below a threshold) and transitions the system to an emergency state (1000 or 0100). On the other hand, in the normal state (0010), the occurrence of a single fault results in state (1010) or (0110), both of which offer acceptable behavior (no voltage dipping takes place due to the capacitive control being on). However, the occurrence of a second fault in one of these two alert states causes the system to transition to an in-extremis state (1110), where the system loses stability. Similarly, the occurrence of a second fault from one of the emergency states (1000 or 0100) causes the system to also lose stability, transiting it to an in-extremis state (1100). In an in-extremis state, system has lost stability and if the power control (generation/load shut-down) is not exercised in a timely fashion (before the “critical fault clearance” time), the system reaches the Failed state. Otherwise (if the power control is exercised in a timely fashion), the system acquires stability and safety but its performance is degraded (some generators/loads are tripped/shed). Thus a recovery state corresponds to a state where power control is on, i.e., \( s_4 = 1 \). A combination of repair of faulted lines and re-energization of generators/loads causes the power system to recover to its normal operating mode.

In Figure 5, the overall plant model \( G \) has 17 states, and the nonfaulty part \( G^N \) consists of the states where no fault has occurred, i.e., the states with label (00--), which includes the normal state (0010), one of the alert states (0000), and two recovery states (0001, and 0011). The model for \( G^N \) is shown...
form of fault-tolerance. It turns out that this weaker form of fault-tolerance can be expressed as a second type of stability property, namely the language-stability property. The notion of weak fault-tolerance is captured as follows:

**Definition 3:** Given a plant \( G \) with its nonfaulty part \( G^N \), \((G, G^N)\) is said to be weakly fault-tolerant if exists \( m \in \mathcal{N} \) such that for \( s \in L(G) - L(G^N) \), \( st \in L(G) \) with \( |t| \geq m \) or \( st \) deadlocks, exists \( u \in L(G^N) \) and \( t' \leq t \) with \( |t'| \leq m \) and \( st' \subseteq_G u \). In this case, \( m \) is called the delay-bound of weakly fault-tolerance.

The following example illustrates a system that is weakly fault-tolerant but not fault-tolerant.

**Example 3:** Let us reconsider \((G, G^N)\) shown in Figure 2, which as we discussed above is not fault-tolerant (due to the presence of the cycle in the faulty part). However after the transition on \( f \), the state 3 is reached, from where all executable traces are also executable starting from the nonfaulty state 2, i.e., the faulty state 3 is simulated by the nonfaulty state 2. To see this consider the behaviors following a faulty trace \(saf\), where \( s \in (c^* + (ab)^*)^*\). Then we claim that those behaviors are also possible following the nonfaulty trace \(sa\), i.e., \(saf \subseteq_G sa\). The reason being that following the execution of \(sa\), the system ultimately executes a trace in \((ba)^*bc\) which brings the system to state 2, whereas the execution of any trace in \((ba)^*bc\) following the trace \(sa\) also brings the system to state 2.

The above notion of weak fault-tolerance can be regarded as a type of language convergence property, as demonstrated by the next theorem. The basic idea is that the post-fault behavior, in a finite number of steps, matches a behavior that is a possible prior to the occurrence of a fault. In other words, the behaviors executable after a post-fault behavior are also executable after the nonfaulty behaviors. Since the matching of behavior following a finite execution is precisely the notion of language convergence, the weak fault-tolerance property can be described as a language convergence property. The faulty behavior is given by, \(L(G) - L(G^N)\). The post-fault behavior has thus the following two parts, the generated part, \(L(G)\setminus[L(G) - L(G^N)]\), and the marked part, \(L_m(G)\setminus[L(G) - L(G^N)]\). Similarly the behavior after no fault has occurred also has two parts, namely \(L(G)\setminus L(G^N)\) and \(L_m(G)\setminus L(G^N)\). For the plant to be weakly fault-tolerant, traces in generated (resp., marked) post-fault behavior should converge to traces in generated (resp., marked) behavior prior to the occurrence of a fault. This is captured in the following theorem.

**Theorem 3:** Given a plant \( G \) and its nonfaulty part \( G^N \), \((G, G^N)\) is weakly fault-tolerant if and only if \(L(G)\setminus[L(G) - L(G^N)]\) converges to \(L(G)\setminus L(G^N)\) and \(L_m(G)\setminus[L(G) - L(G^N)]\) converges to \(L_m(G)\setminus L(G^N)\).

**Proof:** First we show that if \((G, G^N)\) is weakly fault-tolerant, then there exists \(m \in \mathcal{N} \) such that for each \( t \in L(G)\setminus[L(G) - L(G^N)] \) with either \(|t| \geq m \) or \( t \) deadlocks, there exists \( t' \leq t \) and \( v \in L(G)\setminus L(G^N) \) such that \( t = tv \) and \(|v| \leq m \). We claim that \( m \) can be chosen to be the delay-bound of weak fault-tolerance.

Since \( t \in L(G)\setminus[L(G) - L(G^N)] \), there exists \( s \in L(G) - L(G^N) \) such that \( st \in L(G) \). Further since \(|t| \geq m \) or \( t \) holds separately in Figure 6. The specification \( K \) for the overall plant excludes all traces that reach the Failed state, i.e., the Failed state is deemed forbidden for the overall plant. Among the four states of \( G^N \), the normal state \((0010)\) is deemed a final state, i.e., \(K^N\) excludes all traces of \( G^N \) that end at a non-final state. Clearly, \(K^N \subseteq K\).

A fault-tolerant supervisor can be shown to exist and the controlled system under such a supervisor is shown in Figure 7. Each state is labeled with the maximal number of steps it takes to reach a state of \( G^N \). As one can see, the system recovers in no more than 4 steps, i.e, the delay-bound of recovery is 4 steps.

**VI. WEAKLY FAULT-TOLERANT SUPERVISORY CONTROL**

In certain applications, a weaker form of recovery may suffice, namely, the recovery should cause the system to reach a state which is “simulated” by a nonfaulty state. That is, behaviors starting from a state after recovery be subsumed by those starting from a nonfaulty state. Then any safety and liveness property that the system satisfies starting from such a nonfaulty state is also satisfied by the system starting from the state after recovery. This motivates us to introduce a weaker
deadlocks, from fault-tolerance of \((G, G_N)\), there exists \(u \in L(G_N)\) and \(t = t'v\) \((|t'| \leq m \text{ and } st't' \subseteq_G u)\). Since \(st't'v \in L(G)\) and \(u \in L(G_N)\), it follows that \(uv \in L(G)\), therefore \(v \in L(G)|L(G_N)\), as desired.

Similarly, it can be shown that if \((G, G_N)\) is weakly fault-tolerant with delay-bound of fault-tolerance \(m\), then \(t \in L_m(G) \setminus (G_N)\) with \(|t| \geq m\) or \(t\) deadlocks, implies \(t = t'v\) \((|t'| \leq m\) and \(v \in L_m(G)\) \(\subseteq G\)). As above there exists \(s \in L(G) \setminus L(G_N)\) such that \(st \in L(G)\). Invoking the fault-tolerance property and noting that \(st \in L_m(G)\), we can conclude that \(t = t'v\) such that \(|t'| \leq m\) and \(uv \in L_m(G)\) for some \(u \in L(G)\). Since \(u \in L(G)\) and \(uv \in L_m(G)\), it follows that \(v \in L(G) \setminus L(G_N)\), as desired.

In order to prove the sufficiency, we show that the delay-bound \(m\) of fault-tolerance can be chosen to be the same as the delay-bound of convergence of \(L(G) \setminus (G_N)\) to \(L(G)|L(G_N)\) and of \(L_m(G) \setminus (G_N)\) to \(L_m(G)\).

For \(s \in L(G) \setminus L(G_N)\) pick an extension \(t\) such that \(st \in L(G)\) and either \(|t| \geq m\) or \(s\) deadlocks. Then \(t \in L(G) \setminus L(G_N)\) and from the convergence of \(L(G) \setminus (G_N)\) to \(L(G)|L(G_N)\), we have \(t = t'v\) with \(|t'| \leq m\) and \(v \in L(G)\) \(\subseteq G_N\). The latter further implies the existence of \(u \in L(G)\) such that \(uv \in L(G)\). Therefore, \(L(G) \setminus st' \subseteq L(G)\), as desired.

It remains to show that if \(st \in L_m(G)\), then \(uv \in L_m(G)\). This requires the second convergence property. First note that \(st \in L_m(G)\) and \(s \in L(G) \setminus L(G_N)\) implies \(t \in L_m(G) \setminus (G_N)\). So from convergence of \(L_m(G) \setminus (G_N)\) to \(L_m(G)\), it follows that \(t = t'v\) with \(|t'| \leq m\) and \(v \in L_m(G)|L(G_N)\). \(v \in L_m(G)\) further implies the existence of \(u \in L(G)\) such that \(uv \in L(G)\). Therefore, \(L_m(G) \setminus st' \subseteq L_m(G)\), as desired.

A given plant \((G, G_N)\) may not be intrinsically weakly fault-tolerant but could be made so through the use of control. This motivates us to formulate the notion of a weakly fault-tolerant supervisor, which exercises appropriate control actions so that the controlled plant \((G)[S, G^{N}]|S\) is weakly fault-tolerant. The control actions of a weakly fault-tolerant supervisor ensure that following any fault, a recovery takes place within a bounded number of states, i.e., the controlled plant reaches a state, starting from where the executable behaviors are also executable starting from a nonfaulty state.

Definition 4: Given a plant \(G\) with its nonfaulty part \(G_N\), a supervisor \(S\) is said to be weakly fault-tolerant if \((G)[S, G^{N}]|S\) is weakly fault-tolerant.

The following corollary follows from Theorem 3.

Corollary 3: Given a plant \(G\) and its nonfaulty part \(G_N\), a supervisor \(S\) is weakly fault-tolerant if and only if \(L(G)[S] \subseteq L(G)|S\) converges to \(L(G)[S] \subseteq L(G)|S\), and \(L_m(G)[S] \subseteq L_m(G)|S\) converges to \(L_m(G)[S] \subseteq L_m(G)|S\).

VII. EXISTENCE OF WEAKLY FAULT-TOLERANT SUPERVISOR

Using the results of the previous section, we next present a condition for the existence of weakly fault-tolerant super-visor. As before, the supervisor is required to impose a specification \(K\) on the nonfaulty plant \(G_N\) and a possibly more liberal specification \(K \supseteq K_N\) on the overall plant. We have the following existence result.

Theorem 4: Given a plant \(G = (X, \Sigma, \alpha, x_0, X_m)\) with nonfaulty part \(G_N = (X, \Sigma, \alpha'^N, x_0, X^{N}_m)\), specification \(\emptyset \neq K \subseteq L_m(G)\) for \(G\) and specification \(\emptyset \neq K_N \subseteq L_m(G)\) for \(G_N\) satisfying \(K \subseteq K_N\), there exists a nonmarking, nonblocking (with respect to both \(G\) and \(G_N\)), \((\Sigma_u, M)\)-compatible and weakly fault-tolerant supervisor \(S\) such that

1) \(L_m(G)[S] = K\), \(L_m(G_N)[S] = pr(L_m(G_N)[S])\), and
2) \(L_m(G)[S] = K\) and \(L_m(G)[S] = pr(L_m(G)[S])\), if and only if

1) \(K\) is relative-closed, controllable and observable with respect to \(G\),
2) \(pr(K) \setminus (pr(K)\setminus pr(K_N))\) converges to \(pr(K)\setminus pr(K_N)\), and \((K\setminus pr(K)\setminus pr(K_N))\) converges to \((\emptyset\setminus pr(K_N))\),
3) \(K = K \cap L_m(G)\), and \(pr(K_N) = pr(K) \cap L_m(G)\).

Proof: From [17], we know there exists a \((\Sigma_u, M)\)-compatible, nonmarking and nonblocking supervisor \(S\) such that \(L_m(G)[S] = K\) and \(L_m(G)[S] = pr(K_N)\) if and only if \(K\) is relative-closed, controllable and observable with respect to \(G\). Also from Theorem 1, we know \(S\) is weakly fault-tolerant such that \(L_m(G||S) = K\), \(L_m(G_N||S) = pr(K_N)\), \(L_m(G)[S] = K\) and \(L_m(G)[S] = pr(K_N)\) if and only if \(pr(K)\setminus (pr(K)\setminus pr(K_N))\) converges to \(pr(K)\setminus pr(K_N)\) and \((\emptyset\setminus pr(K)\setminus pr(K_N))\) converges to \((\emptyset\setminus pr(K_N))\).

So we only need to show given \(L_m(G)[S] = K\) and \(L_m(G)[S] = pr(K)\), we have \(L_m(G||S) = K\) and \(L_m(G||S) = pr(K)\) if and only if \(K\) and \(K_N\) are constrained by \(K = K \cap L_m(G)\) and \(pr(K_N) = pr(K) \cap L_m(G)\). This is something we proved as part of the proof of Theorem 2.

Remark 2: The complexity of checking conditions 1 and 3 of Theorem 4 are discussed in Remark 1. Here we only discuss the complexity of checking the second condition. The languages appearing in the second condition can be recognized using automata of size \(O(|Q|)\). This is because the automaton \((Q, \Sigma, \delta, Q - Q^N, Q_m)\) (i.e., \(R\) with its initial state replaced by the set \(Q - Q^N\) generates the language \(pr(K)\setminus (pr(K)\setminus pr(K_N))\) and marks the language \((\emptyset\setminus pr(K)\setminus pr(K_N))\), whereas the automaton \((Q, \Sigma, \delta, Q^N, Q_m)\) (i.e., \(R\) with its initial state replaced by the set \(Q^N\)) generates the language \((\emptyset\setminus pr(K)\setminus pr(K_N))\) and marks the language \(K'\setminus pr(K)\) .

Note that the automata \((Q, \Sigma, \delta, Q - Q^N, Q_m)\) and \((Q, \Sigma, \delta, Q^N, Q_m)\) are nondeterministic due to the nonuniqueness of the initial state. In order to verify the language convergence properties of Condition 2, the algorithm given in [29] can be adapted to the nondeterministic setting to verify whether the language \(K'\setminus pr(K)\setminus pr(K_N)\) is equal to the language \(K'\setminus pr(K)\setminus pr(K_N)\). We construct the automaton \(T := (Z, \Sigma, \gamma, Z_0, Z_m)\), where

\[ Z = Q \times 2^Q, \]
\[ Z_0 = \{(q, Q^N)\mid q \in Q - Q^N\}, \]
\[ Z_m = \{(q, Q)\mid q \in Q_m, \hat{Q} \cap Q_m \neq \emptyset\}, \]
Let $\mathcal{Z} := \{(q, \bar{Q}) \mid L_m(R, q) \subseteq L_m(R, \bar{Q})\}$, where $L_m(R, q)$ is the language marked by $R$ starting from the state $q$ and $L_m(R, \bar{Q}) := \bigcup_{q \in \bar{Q}} L_m(R, \bar{q})$. Note $L_m(T) = L_m((Q, \Sigma, \delta, Q - Q^N, Q_m)) = K \setminus pr(K) - pr(K^N)$, and a trace $t \in L_m(T)$ ends at a state $(q, \bar{Q}) \in \mathcal{Z}$ if and only if $t$ possesses a suffix $v \in L_m((Q, \Sigma, \delta, Q^N, Q_m)) = K \setminus pr(K^N)$. Then following the results in [29] it can be shown that $K \setminus pr(K) - pr(K^N)$ converges to $K \setminus pr(K^N)$ if and only if $\mathcal{Z} \subseteq \Omega(\mathcal{Z})$. Next by simply replacing $Q_m$ with $Q$ in the convergence test just described, we can obtain a test for verifying whether $pr(K) \setminus pr(K) - pr(K^N)$ converges to $pr(K) \setminus pr(K^N)$. The complexity of checking the language convergence properties of Condition 2 can accordingly concluded to be $O(|Q|^2(2|Q|^2))$.

The following corollary is a specialization of Theorem 4, where the specification $K$ for the overall plant is simply a safety specification (so that $K = pr(K)$).

**Corollary 4:** Given a plant $G = (X, \Sigma, \alpha, x_0, X_m)$ with nonfaulty part $G^N = (X^N, \Sigma, \alpha, x_0, X_m^N)$, specification $\emptyset \neq K \subseteq L_m(G)$ for $G$ and specification $\emptyset \neq K^N \subseteq L_m(G^N)$ for $G^N$ satisfying $K \subseteq X_m$ is nonmarking, nonblocking, $(\Sigma_m, M)$-compatible and weakly fault-tolerant supervisor $S$ such that

1. $L_m(G_N || S) = K^N$, $L(G^N || S) = pr(L_m(G^N || S))$, and
2. $L(G || S) = K$.

if and only if

1. $K$ is prefix-closed, controllable and observable with respect to $G$,
2. $K \setminus pr(K^N)$ converges to $K \setminus pr(K^N)$,
3. $K^N = K \cap L_m(G^N)$, and $pr(K^N) = K \cap L(G^N)$.

**VIII. Application Example (Contd.)**

In order to illustrate weakly fault-tolerant control we revisit the power system considered in Section V. We relax one of the assumptions that a fault cannot occur in one of the recovery states. The corresponding revised model of the power system is shown in Figure 8. Note the revised model has extra fault transitions.

It can be verified that reaching the recovery state 1101 or 1111 cannot be avoided starting from a nonfaulty state. If the initial nonfaulty state is the normal state or the alert state, then an in-extremis state 1110 or 1100 is reached uncontrollably (through a sequence of two faults), from where the control action $p_0$ must be executed to ensure the unreachability of the failed state causing the system to reach either 1101 or 1111. On the other hand if the initial nonfaulty state is a recovery state (0001 or 0011), the recovery state 1101 or 1111 is reached uncontrollably (through a sequence of two faults).

Consider any of the two recovery states, say 1101. If a controller disables both the repair events at this state, then the system deadlocks and recovery to the nonfaulty part does not occur. Similarly if one of the recovery events is not disabled, the state 1101 becomes part of a cycle of faulty states and so a bounded-delay recovery to nonfaulty states is not guaranteed. It follows that there does not exists a control so that the controlled system is fault-tolerant.

It turns out that there exists a control so that the controlled system is weakly fault-tolerant. Such a controlled system is shown in Figure 9. The controller disables the $c$ and $\bar{c}$ transitions between the emergency and the alert states, and also all the $\bar{p}$ transitions. Due to the presence of the cycles between the recovery states of the faulty part, the controlled system is not fault-tolerant. However the controlled system is weakly fault-tolerant since the traces executable from a recovery state in the faulty part (state 1001 or 1101 or 0101 or 1011 or 1111 or 0111) are also executable from one of the recovery states of the nonfaulty part (state 0001 or 0011), and from any faulty state a recovery state of the faulty part is reached within a bounded-delay.
IX. NONUNIFORMLY BOUNDED FAULT-TOLERANCE

The notion of fault-tolerance we proposed requires the recovery to occur within a uniformly bounded delay. A relaxed notion of fault-tolerance is one where the delay bound for recovery is finite but not necessarily uniformly bounded over all faulty traces. The following definition formalizes the notions of nonuniformly-bounded fault-tolerance.

Definition 5: Given a plant $G$ with its nonfaulty part $G^N$, $(G, G^N)$ is said to be nonuniformly-bounded fault-tolerant if for every $s \in L(G) - L(G^N)$, there exists $m \in N$ such that for $st \in L(G)$ with $|t| \geq m$ or $st$ deadlocks, $u \in L(G^N)$ and $t' \leq t$ with $|t'| < m$ and $st' \subseteq G u$. $(G, G^N)$ is said to be nonuniformly-bounded weakly fault-tolerant if for $s \in L(G) - L(G^N)$, exists $m \in N$ such that for $st \in L(G)$ with $|t| \geq m$ or $st$ deadlocks, $u \in L(G^N)$ and $t' \leq t$ with $|t'| < m$ and $st' \subseteq G u$.

Note in the above definition the delay bound $m$ of fault-tolerance is a function of the faulty trace $s \in L(G) - L(G^N)$. While this is more general than the uniformly-bounded case considered earlier, we show below that when $G$ and $G^N$ are finite-automata, the two notions are equivalent.

Theorem 5: Suppose the plant $G$ and its nonfaulty part $G^N$ are finite automata. Then

1) $(G, G^N)$ is nonuniformly-bounded fault-tolerant if and only if $(G, G^N)$ is fault-tolerant.

2) $(G, G^N)$ is nonuniformly-bounded weakly fault-tolerant if and only if $(G, G^N)$ is weakly fault-tolerant.

Proof: We begin by proving the first part. Without loss of generality $G$ is assumed to be minimal. We show that when $(G, G^N)$ is nonuniformly-bounded fault-tolerant, it holds that $X - X^N \subseteq \Omega (X^N)$ (which from Theorem 1 is equivalent to $(G, G^N)$ being fault-tolerant). Suppose for contradiction that this is not true. Then either there exists a cycle of states in $X - X^N$ or some state in $X - X^N$ is a deadlocking state. Let $s \in L(G) - L(G^N)$ be a trace that ends on such a cycle or at such a deadlocking state. In the former case (when $s$ ends on a cycle), for every $m \in N$ there exists an extension $st \in L(G)$ along the cycle such that all states visited beyond the trace $s$ belong to $X - X^N$. From minimality of $G$, exists $t' \leq t$ such that $st' \subseteq G u$ for some $u \in L(G^N)$, contradicting the fact that $(G, G^N)$ is nonuniformly-bounded fault-tolerant. The same conclusion is obtained in the latter case (when $s$ ends at a deadlocking state). This completes the proof of the first part.

To prove the second part, we consider the automaton $T := (Z, \Sigma, \gamma, Z_0, Z_m)$ constructed in Remark 2 and show that if $(G, G^N)$ is nonuniformly-bounded weakly fault-tolerant, then $Z \subseteq \Omega (\hat{Z})$, which as mentioned in Remark 2 is equivalent to the fact that $L_m(G) \setminus [L(G) - L(G^N)]$ converges to $L_m(G) \setminus L(G^N)$. (The convergence of $L(G) \setminus [L(G) - L(G^N)]$ to $L(G) \setminus L(G^N)$ can be proved similarly by setting $Q_m = Q$ in the definition of the automaton $T = (Z, \Sigma, \gamma, Z_0, Z_m)$ and establishing that $Z \subseteq \Omega (\hat{Z})$.) Note together these two language convergence properties are equivalent to $(G, G^N)$ being weakly fault-tolerant (see Theorem 3).

Suppose for contradiction that $Z \not\subseteq \Omega (\hat{Z})$. Then either there exists a cycle of states belonging to $Z - \hat{Z}$ or some state in $Z - \hat{Z}$ is a deadlocking state. In the former case, for any $m \in N$, exists trace $t \in L_m(T) = L_m((Q, \Sigma, \delta, Q - Q^N, Q_m)) = L_m(G) \setminus [L(G) - L(G^N)]$ with $|t| \geq m$ such that no suffix of $t$ belongs to $L_m((Q, \Sigma, \delta, Q^N, Q_m)) = L_m(G) \setminus L(G^N)$. So exists $s \in L(G) - L(G^N)$ such that $st \in L(G)$ and $t$ is arbitrarily long, yet there do not exist suffix $v$ of $t$ and trace $u \in L(G^N)$ such that $wv \in L_n(G)$. This is a contradiction to the fact that $(G, G^N)$ is nonuniformly-bounded weakly fault-tolerant. The same conclusion can be arrived at even in the latter case when $Z - \hat{Z}$ possesses a deadlocking state. This completes the proof.

The following example shows that in general nonuniformly-bounded fault-tolerance is weaker than the uniformly-bounded case.

Example 4: Consider a plant $G$ and its nonfaulty part $G^N$ shown in Figure 10 with $L(G) = \cup_{n \geq 1} pr(a^n f b^n)$, $L_m(G) = \{\epsilon\} \cup_{n \geq 1} a^n f b^n$, $L(G^N) = a^*$, and $L_m(G^N) = \emptyset$, where $f$ represents the faulty event. Then $L(G) - L(G^N) = \cup_{n \geq 2} a^n f b^n$. Pick $s_n = a^n f \in L(G) - L(G^N)$. Then $L_m(G) \setminus \{s_n\} \subseteq L_m(G) \setminus [L(G) - L(G^N)]$. The only suffix of $b^n$ that is equivalent to a trace in $L(G^N) = a^*$ is the $\epsilon$ trace. So the delay-bound of fault-tolerance for the trace $s_n$ is given by $n$ (and is bounded). However this delay-bound grows unboundedly as the index $n$ of $s_n$ grows. We conclude that $(G, G^N)$ is nonuniformly-bounded fault-tolerant, but it is not fault-tolerant.

![Fig. 10. $(G, G^N)$ that is only nonuniformly bounded fault-tolerant](image)

X. CONCLUSION

We presented a framework for fault-tolerant supervisory control. Notations of fault-tolerance and weakly fault-tolerance have been proposed. Given a plant along with its nonfaulty part, the goal of a fault-tolerant supervisory control is to enforce a specification for the nonfaulty plant and another (perhaps more liberal) specification for the overall plant, and also to ensure a bounded delay recovery up on the occurrence of a fault. Recovery implies that the ensuing behaviors are equivalent to those starting from a nonfaulty state. In case of weak fault-tolerance, recovery implies that the ensuing behaviors are subsumed by those starting from a nonfaulty state. Necessary and sufficient conditions for the existence of fault-tolerant as well as weakly fault-tolerant supervisor are provided. The condition involves the usual notions of controllability, observability and relative-closure, together with the notion of stability. The notion of state-stability is needed for fault-tolerance, whereas the weak fault-tolerance requires the notion of language-stability. Algorithms to verify state-stability are presented in [7], [24] and are of linear complexity. Algorithms to verify language-stability are presented.
in [18], [29]; the complexity is polynomial in the plant language (the language which needs to converge) and quadratic-exponential in the specification language (the language to which the convergence occurs). We also introduced the notion of nonuniformly-bounded fault-tolerance (and its weak version) where the delay-bound for recovery is not uniformly bounded over the set of faulty traces, and showed that this notion is equivalent to the notion of “uniformly-bounded fault-tolerance” considered earlier when the underlying system is one of finitely many states. Future work will explore the synthesis of maximally-permissive fault-tolerant supervisors.

REFERENCES


Qin Wen (S’04) received the B.Sc. and M.Sc. degrees from University of Science and Technology of China(USTC), Hefei, China, in 2001 and 2004, respectively. He is currently a Ph.D. candidate in the Department of Electrical and Computer Engineering at Iowa State University, Ames, IA. His research interests include modeling, specification, verification, and control of event-driven, real-time, fault-tolerant, and embedded systems, formal verification and formal methods, etc.

Ratnesh Kumar (F’07) received the B.Tech. degree in Electrical Engineering from the Indian Institute of Technology at Kanpur, India, in 1987, and the M.S. and the Ph.D. degree in Electrical and Computer Engineering from the University of Texas at Austin, Texas, in 1990 and 1991, respectively. From 1991-2002 he was on the faculty of University of Kentucky, and since 2002 he is on the faculty of the Iowa State University. He has held visiting position at the Institute of Systems Research at the University of Maryland at College Park, the Applied Research Laboratory at the Pennsylvania State University, the NASA Ames Research Center, and the Argonne National Laboratory-West. His primary research interest is Modeling, Verification, Control, and Diagnosis of Reactive/Event-driven, Real-time, and Hybrid Systems and their applications.

He was a recipient of the Microelectronics and Computer Development (MCD) Fellowship from the University of Texas at Austin, and was awarded the Lalit Narain Das Memorial Gold Medal for the Best EE Student and the Ratan Swarup Memorial Gold Medal for the Best All-rounder Student from the Indian Institute of Technology at Kanpur, India. He is a recipient of the NSF Research Initiation Award, NASA-ASEE summer faculty fellowship award, and coauthor of the book Modeling and Control of Logical Discrete Event Systems, Kluwer Academic Publishers, 1995. He serves on the program committee for the IEEE Control Systems Society, and the International Workshop on Discrete Event Systems. He is an associate editor of SIAM Journal on Control and Optimization, Journal of Discrete Event Dynamical Systems and IEEE Control Systems Society and is a past editor of IEEE Transactions on Robotics and Automation.
**Jing Huang** (S’02) received his B.Sc. and M.Sc. degrees from Xiamen University, Xiamen, China, in 1997 and 2000, respectively. He also received a M.Sc. degree from The University of Texas at Dallas, Richardson, TX in 2003. He is currently a Ph.D. candidate in the Department of Electrical and Computer Engineering at Iowa State University, Ames, IA. His research interests include modeling, specification, verification, and control of event-driven, real-time, fault-tolerant, and embedded systems, formal verification and testing of hardware and software systems, sensor networks, formal methods, etc.

**Haifeng Liu** (S’04-M’08) received the B.S. and M.S. degrees in electrical engineering from Zhejiang University, Hangzhou, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering from Iowa State University, Ames, Iowa, in 2007. Currently, he is with the Department of Regional Transmission-South at California Independent System Operator.