DECENTRALIZED DIAGNOSIS OF EVENT-DRIVEN SYSTEMS FOR SAFELY REACTING TO FAILURES

Wenbin Qiu and Ratnesh Kumar

Dept. of Elec. & Comp. Eng., Iowa State University,
Ames, IA 50011, U.S.A.
{wqiu,rkumar}@iastate.edu

Abstract: We introduce the notion of *safe-codiagnosability*, extending the notion of safe-diagnosability (Paoli and Lafortune, 2003) to the decentralized setting, where there exist multiple diagnosers performing diagnosis using their own observations without communicating to each other. For a system, a certain sub-behavior is deemed safe (captured via a safety specification), and a further sub-behavior is deemed non-faulty (captured via a non-fault specification). Safe-codiagnosability requires that when the system executes a trace that is faulty, there exists at least one diagnoser that can detect this within bounded delay and also before the safety specification is violated. The above notion of safe-codiagnosability may also be viewed as an extension of the notion of codiagnosability (Qiu and Kumar, 2004), where the latter did not have any safety requirement. We show that safe-codiagnosability is equivalent to codiagnosability together with "zero-delay codiagnosability" of "boundary safe traces". (A safe trace is a boundary safe trace, if exists a single-event extension that is unsafe.) We give an algorithm of polynomial complexity for verifying safe-codiagnosability. For a safe-codiagnosable system, the same methods as those proposed in (Qiu and Kumar, 2004) can be applied for off-line synthesis of individual diagnosers, as well as for on-line diagnosis using them.

Keywords: Discrete-event systems, Fault diagnosis, Safety analysis, Decentralized systems, Automata theory

1. INTRODUCTION

Failure diagnosis is an active area of research, and has received considerable attention in the literature. A failure is a deviation from an expected or desired behavior. Various approaches have been proposed for failure diagnosis, including fault-trees, expert systems, neural networks, fuzzy logic, Bayesian networks, and analytical redundancy These are broadly categorized into non-model based (where observed behavior is matched to known failures), and model based (where observed behavior is compared against model predictions for any abnormality).

For discrete event systems (DESs) – systems with discrete states that change when certain events occur, a certain model based approach for failure diagnosis is proposed in (Sampath *et al.*, 1995). The property of *diagnosability* requires that once

[1] The research was supported in part by the National Science Foundation under the grants NSF-ECS-0099851, NSF-ECS-0218207, NSF-ECS-0244732, NSF-EPNES-0323379, and NSF-ECS-0424048, and a DoD-EPSCoR grant through the Office of Naval Research under the grant N000140110621.

a failure has occurred, it be detected and diagnosed within bounded "delay" (within bounded number of transitions). The diagnosability can be tested polynomially as shown later in (Jiang *et al.*, 2001; Yoo and Lafortune, 2002). In (Das and Holloway, 2000; Pandalai and Holloway, 2000), a template based approach was developed for failure diagnosis in timed discrete event system. The above approaches can be thought to be "event-based" as failure is modeled as execution of certain "faulty events". An equivalent "state-based" approach was considered in (Lin, 1994), where the occurrence of a failure is modeled as reaching of certain "faulty states". To facilitate generalization of failure specifications, linear-time temporal logic (LTL) based specification and diagnosis of its failure was proposed in (Jiang and Kumar, 2004). A theory for failure diagnosis of repeatedly-occurring/intermittent failures was introduced in (Jiang *et al.*, 2003).

The above mentioned work dealt with *centralized* failure diagnosis, where a central diagnoser is responsible for failure detection and diagnosis in the system. (Debouk *et al.*, 2000) addressed the problem of distributed failure diagnosis based on a "coordinated decentralized architecture", where local diagnosers do not communicate with each other directly, but send local information to a coordinator. Then the coordinator makes the final diagnosis decision. (Sengupta and Tripakis, 2002) discussed the distributed diagnosis problem, where communication directly exists between local diagnosers, and is assumed to be lossless, and in order. Notion of "decentralized diagnosis" was formulated, which was proved to be undecidable. It is now understood that decentralized diagnosis is not strong enough to capture distributed diagnosability under unbounded delay communication; a required stronger notion is introduced in (Qiu *et al.*, 2004), and further this new notion is shown to be decidable. The decentralized diagnosis problem with asymmetric communication was discussed in (Boel and van Schuppen, 2002), where communication is one-way and without delays. In a prior work (Qiu and Kumar, 2004), we studied the problem of decentralized failure diagnosis, where the system failure is diagnosed by multiple local diagnosers. A notion of *codiagnosability* was introduced to capture the fact that the occurrence of any failure must be diagnosed within bounded delay by at least one local diagnoser using its own observations of the system execution. Polynomial algorithms were provided for (i) testing codiagnosability, (ii) computing the delay bound of diagnosis, (iii) off-line synthesis of diagnosers, and (iv) on-line diagnosis using them.

In order to react to a failure in a timely fashion, while it is necessary that the failure be detected within a bounded delay, such a property alone is not sufficient. It is also needed that the detection occur before the system behavior becomes "unsafe". To capture this additional requirement for failure detection, the notion of *safe-diagnosability* was introduced in (Paoli and Lafortune, 2003). We extend this notion to the decentralized setting, where there exist multiple diagnosers performing diagnosis using their own observations without communicating to each other, by formulating the notion of *safe-codiagnosability*. For a system, a certain sub-behavior is deemed safe (captured via a *safety specification*), and a further sub-behavior is deemed non-faulty (captured via a *non-fault specification*). The safe behavior includes all of non-faulty behavior and some of post-fault behavior where system performance may be degraded but still tolerable. Safe-codiagnosability requires that when the system executes a trace that is faulty, then exists at least one diagnoser that can detect this within bounded delay and also before the safety specification is violated. The above notion of safe-codiagnosability may also be viewed as an extension of the notion of codiagnosability (Qiu and Kumar, 2004), where the latter did not have any safety requirement. We show that safe-codiagnosability is equivalent to codiagnosability together with "zero-delay codiagnosability" of "boundary safe traces". (A safe trace is a boundary safe trace, if exists a single-event extension that is unsafe.) We give an algorithms of polynomial complexity for verifying safe-codiagnosability. (The verification algorithm presented in (Paoli and Lafortune, 2003) was based upon the structural property of a deterministic diagnoser, and had an exponential complexity owing to the exponential size of the diagnoser.) For a safe-codiagnosable system, the same methods as those proposed in (Qiu and Kumar, 2004) can be applied for off-line synthesis of individual diagnosers, as well as for on-line diagnosis using them.

2. NOTIONS AND PRELIMINARIES

Given an event set Σ, Σ^* is used to denote the set of all finite length event sequences over Σ, including the zero length event sequence ϵ. A member of Σ^* is a *trace* and a subset of Σ^* is a *language*. Given a language $K \subseteq \Sigma^*$, the *complement* of K, denoted $K^c \subseteq \Sigma^*$, is defined as $K^c := \Sigma^* - K$. If trace s is a *prefix* of trace t, it is denoted as $s \leq t$. Given a language $K \subseteq \Sigma^*$, its prefix-closure, denoted $pr(K)$, is defined as, $pr(K) := \{s \in \Sigma^* | \exists t \in K \text{ s.t. } s \leq t\}$, and K is said to be *prefix-closed* if $K = pr(K)$. The *supremal prefix-closed sublanguage* of K, denoted $supP(K) \subseteq K$, is defined as, $supP(K) := \{s \in K | pr(s) \subseteq K\}$. The *quotient* of K_1 with respect to K_2 is defined as $K_1/K_2 := \{s \in \Sigma^* | \exists t \in K_2 \text{ s.t. } st \in K_1\}$. The set of *deadlocking traces*

of a language K are those traces from which no further extensions exist in K, i.e., $s \in K$ is deadlocking trace if $\{s\}\Sigma^* \cap K = \{s\}$.

A DES is modeled as a *finite state machine* (FSM)/*finite automaton* (FA) G and is denoted by $G := (X, \Sigma, \alpha, x_0)$, where X is the set of states, Σ is the finite set of events, $x_0 \in X$ is the initial state, and $\alpha : X \times \overline{\Sigma} \to 2^X$ is the transition function, where $\overline{\Sigma} := \Sigma \cup \{\epsilon\}$. G is said to be *deterministic* if $|\alpha(\cdot,\cdot)| \leq 1$ and $|\alpha(\cdot,\epsilon)| = 0$; otherwise, it is called *nondeterministic*. $(x, \sigma, x') \in X \times \overline{\Sigma} \times X$ is a transition of G if $x' \in \alpha(x, \sigma)$; it is an ϵ-transition if $\sigma = \epsilon$. Letting $\epsilon^*(x)$ denote the set of states reachable from x in zero or more ϵ-transitions, the transition function α can be extended from domain $X \times \overline{\Sigma}$ to domain $X \times \Sigma^*$ recursively as follows: $\forall x \in X, s \in \Sigma^*, \sigma \in \Sigma, \alpha(x, \epsilon) = \epsilon^*(x)$, and $\alpha(x, s\sigma) = \epsilon^*(\alpha(\alpha(x, s), \sigma)))$. The *generated language* by G is defined as $L(G) := \{s \in \Sigma^* | \alpha(x_0, s) \neq \emptyset\}$, i.e., it includes all traces that can be executed from the initial state of G. States reached by execution of deadlocking traces in $L(G)$ are called deadlocking states. A *path* in G is a sequence of transitions $(x_1, \sigma_1, x_2, \cdots, \sigma_{n-1}, x_n)$, where $\sigma_i \in \overline{\Sigma}$ and $x_{i+1} \in \alpha(x_i, \sigma_i)$ for all $i \in \{1, \cdots, n-1\}$. The path is called a *cycle* if $x_1 = x_n$.

Given an automaton $G = \{X, \Sigma, \alpha, x_0\}$, the complete model of G is defined as $\overline{G} = \{\overline{X}, \Sigma, \overline{\alpha}, x_0\}$, where $\overline{X} := X \cup \{F\}$, and $\overline{\alpha}$ is defined as follows. $\forall \overline{x} \in \overline{X}, \sigma \in \Sigma, \overline{\alpha}(\overline{x}, \sigma) :=$

$$\begin{cases} \alpha(\overline{x}, \sigma), & \text{if } [\overline{x} \in X] \wedge [\alpha(\overline{x}, \sigma) \neq \emptyset] \\ F, & \text{if } [\overline{x} = F] \vee [\alpha(\overline{x}, \sigma) = \emptyset] \end{cases}.$$

Since all events are defined at each state, the complete model $\overline{G}$ generates the language Σ^*, i.e., $L(\overline{G}) = \Sigma^*$.

Given two automata $G = (X, \Sigma, \alpha, x_0)$ and $R = (Y, \Sigma, \beta, y_0)$, the *synchronous composition* of G and R is defined as, $G||R = (X \times Y, \Sigma, \gamma, (x_0, y_0))$ such that

$$\forall (x, y) \in X \times Y, \sigma \in \overline{\Sigma}, \gamma((x, y), \sigma) :=$$

$$\begin{cases} \alpha(x, \sigma) \times \beta(y, \sigma), & \text{if } \sigma \neq \epsilon; \\ (\alpha(x, \epsilon) \times \{y\}) \cup (\{x\} \times \beta(y, \epsilon)), & \text{otherwise.} \end{cases}$$

If the system execution is observed through a single global observer, we can define a *global observation mask* as $M : \Sigma \to \overline{\Delta}$, where $\overline{\Delta} := \Delta \cup \{\epsilon\}$ and Δ is the set of observed symbols. The definition of M can be extended from events to event sequences inductively as follows: $M(\epsilon) = \epsilon$; $\forall s \in \Sigma^*, \sigma \in \Sigma, M(s\sigma) = M(s)M(\sigma)$. Given an automaton G and mask M, $M(G)$ is the *masked automaton* of G with each transition (x, σ, x') of G replaced by $(x, M(\sigma), x')$. The *local observation masks* associated with different local observers are defined as $M_i : \Sigma \to \overline{\Delta}_i$ $(i \in I = \{1, \cdots, m\})$, where m is the number of local observers, $\overline{\Delta}_i := \Delta_i \cup \{\epsilon\}$ and Δ_i is the set of locally observed symbols.

3. SAFE-CODIAGNOSABILITY

In this section, we present the definition of safe-codiagnosability and the "separation property" of safe-codiagnosability. As described in (Qiu and Kumar, 2004), *for the purpose of diagnosis, a system with deadlocking states can be converted to a deadlock free system by adding a self-loop labeled ϵ at each of its deadlocking state without affecting the diagnosis analysis.* So without loss of generality, we assume a system to be diagnosed, a "plant", to be deadlock free.

Definition 1. (Qiu and Kumar, 2004) Let L be the prefix-closed language generated by a plant, and K be a prefix-closed sublanguage specifying the non-faulty plant behavior ($K \subseteq L$). Assume there are m local sites with observation masks $M_i : \Sigma \to \overline{\Delta}_i$ $(i \in I = \{1, \cdots, m\})$. (L, K) is said to be *codiagnosable* with respect to $\{M_i\}$ if

$$(\exists n \in \mathcal{N})(\forall s \in L - K)(\forall st \in L - K, |t| \geq n) \Rightarrow$$
$$(\exists i \in I)(\forall u \in M_i^{-1} M_i(st) \cap L, u \in L - K) \quad (1)$$

Lemma 1. Let L and K be prefix-closed plant and non-fault specification languages respectively with $K \subseteq L$, and for $i \in I$, M_i be observation mask of site i. Then (L, K) is codiagnosable with respect to $\{M_i\}$ if and only if

$$\exists n \in \mathcal{N} : [(L - K)\Sigma^{\geq n} \cap L] \bigcap_{i \in I} M_i^{-1} M_i(K) = \emptyset.$$

Remark 1. We can introduce the notion of "zero delay codiagnosability" by setting $n = 0$ in the definition of codiagnosability provided by Lemma 1. Then (L, K) is said to be *zero-delay codiagnosable* with respect to $\{M_i\}$ if

$$(L - K) \bigcap_{i \in I} M_i^{-1} M_i(K) = \emptyset, \quad (2)$$

which is equivalent to $\bigcap_{i \in I} M_i^{-1} M_i(K) \cap L \subseteq K$, i.e., (L, K) is zero-delay codiagnosable if and only if the non-faulty behavior K is *decomposable* (Rudie and Wonham, 1992) with respect to the non-faulty+faulty (plant) behavior L. We say a faulty sublanguage $H \subseteq L - K$ is zero-delay codiagnosable with respect to $\{M_i\}$ if $H \bigcap_{i \in I} M_i^{-1} M_i(K) = \emptyset$.

Definition 1 captures the system property that a failure event can be diagnosed within bounded delay after its occurrence by at least one of the

local sites. In order to react to a failure in a timely fashion, it is also needed that a failure be detected before system behavior becomes "unsafe". Safe behavior includes all of non-faulty behavior and some of post-fault behavior where system performance may be degraded but still tolerable. The safety specification, denoted K_S, is another prefix-closed sublanguage of plant language, containing the non-fault specification, i.e., $K \subseteq K_S \subseteq L$.

Definition 2. Let L be the prefix-closed language generated by a plant, and K and K_S be prefix-closed non-fault and safety specification languages contained in L, respectively ($K \subseteq K_S \subseteq L$). Assume there are m local sites with observation masks $M_i : \Sigma \to \overline{\Delta}_i$ ($i \in I = \{1, \cdots, m\}$). (L, K, K_S) is said to be *safe-codiagnosable* with respect to $\{M_i\}$ if

$$(\exists n \in \mathcal{N})(\forall s \in L-K)(\forall st \in L-K, |t| \geq n) \Rightarrow$$
$$(\exists i \in I)(\exists v \in pr(st) \cap K_S)$$
$$(\forall u \in M_i^{-1}M_i(v) \cap L, u \in L-K) \quad (3)$$

Definition 2 has the following meaning. A system is safe-codiagnosable if there exists a delay bound n such that for all faulty trace $s \in L-K$ and all extension t of s with length longer than delay bound ($|t| \geq n$), there exists a site i and a safe prefix v of st such that for all v-indistinguishable u at site i, u is a faulty trace in $L-K$. Informally, Definition 2 means that for any faulty trace, there exists at least one local site that can unambiguously detect that failure within bounded delay and before safety is violated.

Just as we provided an alternative definition of codiagnosability in Lemma 1, we provide an alternative definition of safe-codiagnosability in the following lemma.

Lemma 2. Let L, K, and K_S be prefix-closed plant, non-fault specification, and safety specification languages respectively, and for $i \in I$, M_i be observation mask of site i. Then (L, K, K_S) is safe-codiagnosable with respect to $\{M_i\}$ if and only if

$$\exists n \in \mathcal{N} : [(L-K)\Sigma^{\geq n} \cap L]$$
$$\cap sup P[\bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^c] = \emptyset.$$

To facilitate the development of a test for safe-codiagnosability, we show that the property of safe-codiagnosability can be separated into codiagnosability together with zero-delay codiagnosability of set of boundary safe traces, where a boundary safe trace is a safe trace for which exists a single-event extension that is unsafe.

Definition 3. Given prefix-closed plant language L and safety specification language K_S, a safe trace $s \in K_S$ is called a *boundary safe trace* if exists $\sigma \in \Sigma$ such that $s\sigma \in L - K_S$, i.e., $s \in [(L-K_S)/\Sigma] \cap K_S$. The set of all boundary safe traces is called the *boundary safe language*, denoted K_S^∂, and is given by $K_S^\partial = [(L-K_S)/\Sigma] \cap K_S$.

Theorem 1. Let L, K and K_S be plant language, non-fault specification language, and safety specification language, respectively. (L, K, K_S) is safe-codiagnosable with respect to $\{M_i\}$ if and only if
1. (L, K) codiagnosable with respect to $\{M_i\}$: $\exists n \in \mathcal{N} : [(L-K)\Sigma^{\geq n} \cap L] \bigcap_{i \in I} M_i^{-1}M_i(K) = \emptyset$;
2. K_S^∂ zero-delay codiagnosable with respect to $\{M_i\}$: $K_S^\partial \bigcap_{i \in I} M_i^{-1}M_i(K) = \emptyset$.

4. VERIFICATION OF SAFE-CODIAGNOSABILITY

The algorithm for verifying safe-codiagnosability is based upon checking whether there exists a situation that violates the conditions of safe-codiagnosability. From Theorem 1, we know that safe-codiagnosability can be verified by checking codiagnosability of (L, K) together with zero-delay codiagnosability of K_S^∂, the set of boundary safe traces.

Algorithm 1. Consider the non-fault specification, and the safety specification models, $G = (X, \Sigma, \alpha, x_0)$, $R = (Y, \Sigma, \beta, y_0)$, and $R_S = (Y_S, \Sigma, \beta_S, y_0^S)$, respectively. The corresponding plant, non-fault specification, and safety specification languages are $L = L(G), K = L(R)$, and $K_S = L(R_S)$, respectively, where $K \subseteq K_S \subseteq L$. Let M_i be the observation mask of site i ($i \in I$). To check the safe-codiagnosability of (L, K, K_S), perform the following steps:

Step 1: Check the codiagnosability of (L, K)

Construct a testing automaton $T = (G \| \overline{R}) \times R \times R$ for verifying the codiagnosability of (L, K). This automaton is defined as $T = (Z, \Sigma_T, \gamma, z_0)$, where

- $Z = (X \times \overline{Y}) \times Y \times Y$.
- $\Sigma_T = \overline{\Sigma}^3$, where $\overline{\Sigma} = \Sigma \cup \{\epsilon\}$.
- $z_0 = ((x_0, y_0), y_0, y_0)$.
- $\gamma : Z \times \overline{\Sigma}^3 \to Z$ is defined as: $\forall z = ((x, y), y_1, y_2) \in Z, \sigma_T = (\sigma, \sigma_1, \sigma_2) \in \Sigma_T - \{(\epsilon, \epsilon, \epsilon)\}$, $\gamma(z, \sigma_T) := ((\alpha(x, \sigma), \overline{\beta}(y, \sigma)), \beta(y_1, \sigma_1), \beta(y_2, \sigma_2))$ if and only if

$$[M_1(\sigma) = M_1(\sigma_1)] \wedge [M_2(\sigma) = M_2(\sigma_2)]$$
$$\wedge[(\alpha(x, \sigma) \neq \emptyset) \vee (\overline{\beta}(y, \sigma) \neq \emptyset) \vee$$
$$(\beta(y_1, \sigma_1) \neq \emptyset) \vee (\beta(y_2, \sigma_2) \neq \emptyset)]$$

Note that the silent-transition ϵ is defined at each state of any automaton as a self loop by

default. The testing automaton T is used to track if exists a triplet of traces s, u_1 and u_2 such that s is a faulty trace ($s \in L - K$), and u_i is a s-indistinguishable non-fault trace under mask M_i ($i \in \{1, 2\}$).

Then check if exists an "offending cycle" $cl_T = (z^k, \sigma_T^k, z^{k+1}, \cdots, z^l, \sigma_T^l, z^k)$ such that $\exists i \in [k, l]$ such that $(\overline{y}^i = F) \wedge (\sigma^i \neq \epsilon)$, where $z^i = ((x^i, \overline{y}^i), y_1^i, y_2^i) \in Z$, and $\sigma_T^i = (\sigma^i, \sigma_1^i, \sigma_2^i) \in \Sigma_T$. If the answer is yes, then (L, K) is not codiagnosable, and (L, K, K_S) is not safe-codiagnosable as well. Otherwise, go to the next step.

Step 2: Compute the set of "boundary safe states" B in $G\|R_S$

Construct the composition $G\|R_S$, and define the set of *boundary safe states* as, $B := \{(x, y_S) \in X \times Y_S | \exists \sigma \in \Sigma : \alpha(x, \sigma) \neq \emptyset, \beta_S(y_S, \sigma) = \emptyset\}$. Note that if $s \in L(G\|R_S) = L(G) \cap L(R_S) = L \cap K_S = K_S$ is such that execution of s results in reaching a state $(x, y_S) \in B$, then exists $\sigma \in \Sigma$ such that $s\sigma \in L - K_S$, i.e., $s \in (L - K_S)/\Sigma$. It follows that $s \in K_S^\partial$.

Step 3: Check the zero-delay codiagnosability of K_S^∂ with respect to $\{M_i\}$

Construct a testing automaton $T_S = (G\|R_S) \times R \times R$ for verifying the zero-delay codiagnosability of K_S^∂, where T_S is obtained by replacing $\overline{R}$ by R_S in the testing automaton T constructed above. Let $T_S = (Z_S, \Sigma_T, \gamma_S, z_0^S)$, where Z_S, γ_S, and z_0^S of T_S are defined similarly as Z, γ, and z_0 of T, respectively (with $\overline{R}$ replaced by R_S). Then check if exists an "offending state" $((x, y_S), y_1, y_2)$ in T_S with $(x, y_S) \in B$. K_S^∂ is zero-delay codiagnosable if and only if the answer is no. If K_S^∂ is zero-delay codiagnosable, then (L, K, K_S) is safe-codiagnosable as well (since (L, K) was determined to be codiagnosable above). Otherwise, (L, K, K_S) is not safe-codiagnosable.

Remark 2. Let $|X|$, $|Y|$ and $|Y_S|$ be the number of states in plant G, non-fault specification R, and safety specification R_S respectively, and $|\Sigma|$ be the number of events. $L = L(G), K = L(R), K_S = L(R_S)$. Assume there are m local sites. It was shown in (Qiu and Kumar, 2004) that the complexity for constructing the testing automaton T and checking codiagnosability of (L, K) is $O(|X| \times |Y|^{m+1} \times |\Sigma|^{m+1})$. Using a similar analysis, we can verify that the complexity for constructing the testing automaton T_S and checking the zero-delay codiagnosability of K_S^∂ is $O(|X| \times |Y_S| \times |Y|^m \times |\Sigma|^{m+1})$. It follows that overall complexity of checking safe-codiagnosability of (L, K, K_S) is, $O(|X| \times (|Y| + |Y_S|) \times |Y|^m \times |\Sigma|^{m+1})$.

Once a system is deemed safe-codiagnosable, the same methods as those presented in (Qiu and Kumar, 2004) can be applied for the synthesis of local diagnosers as well as for on-line diagnosis using them. This is because a diagnoser simply observes the plant behavior and reports a fault when it becomes certain about it. The property of safe-codiagnosability guarantees that at least one diagnoser become certain within bounded delay of the occurrence of a fault and prior to the system behavior becoming unsafe. Details are omitted here.

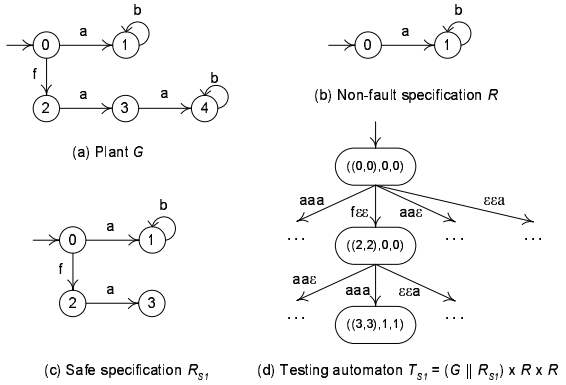

Fig. 1. Models G, R and R_{S1}, and testing automaton T_{S_1} (right)

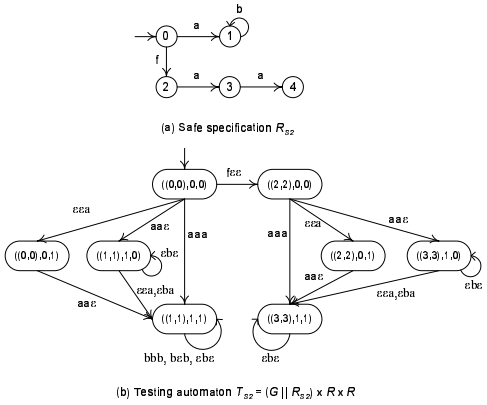

Fig. 2. Safe specification model R_{S2} and testing automaton T_{S_2}

Example 1. Figure 1 (a), (b) and (c) show a plant model G, a non-fault specification model R, and a safety specification model R_S. The set of events is given by $\Sigma = \{a, b, f\}$. There are two local sites, with their observation masks given as follows:

. $M_1(a) = a, M_1(b) = M_1(f) = \epsilon$;
. $M_2(b) = b, M_2(a) = M_2(f) = \epsilon$.

It can be verified that $(L(G), L(R))$ is codiagnosable with respect to $\{M_i\}$ by constructing a testing automaton $T = (G\|\overline{R}) \times R \times R$, which is omitted here.

Since $L = L(G) = pr(ab^* + faab^*)$ and $K_{S_1} = pr(ab^* + fa)$, the boundary safe language $K_{B_1}^\partial = [(L - K_{S_1})/\Sigma] \cap K_{S_1} = \{fa\}$. Following the trace fa, state "3" in G and state "3" in R_{S_1} are reached. Thus, the set of boundary safe states is

given by, $B_1 = \{(3,3)\}$. Figure 1 (d) shows a part of the testing automaton $T_{S_1} = (G \| R_{S_1}) \times R \times R$, where an offending state $((3,3),1,1)$ is reached. Therefore, $K^{\partial}_{B_1}$ is not zero-delay codiagnosable with respect to $\{M_i\}$, and thus (L, K, K_{S_1}) is not safe-codiagnosable with respect to $\{M_i\}$ as well.

Now, if we relax the safety requirement by considering a new enlarged safety specification model R_{S_2} as shown in Figure 2 (a), the system becomes safe-codiagnosable. To see this, since $K_{S_2} = pr(ab^* + faa)$, the boundary safe language is given by, $K^{\partial}_{B_2} = [(L - K_{S_2})/\Sigma] \cap K_{S_2} = \{faa\}$. Thus, the set of boundary safe states is given by, $B_2 = \{(4,4)\}$. The new testing automaton $T_{S_2} = (G \| R_{S_2}) \times R \times R$ is shown in Figure 2 (b), where no offending states (states with first pair of coordinates being $(4,4)$) are reached. Therefore, $K^{\partial}_{B_2}$ is zero-delay codiagnosable with respect to $\{M_i\}$, and thus (L, K, K_{S_2}) is safe-codiagnosable with respect to $\{M_i\}$ as well.

5. CONCLUSION

This paper studies the property of being able to react safely to failures in a decentralized setting. For this purpose a notion of safe-codiagnosability is introduced by extending the notion of safe-diagnosability (Paoli and Lafortune, 2003) to the decentralized setting. Safe-codiagnosability captures the property that when a system executes a trace that is faulty, there exists at least one diagnoser that can detect this within bounded delay and also before the system behavior becomes "unsafe". Necessary and sufficient conditions for safe-codiagnosability are established, showing that safe-codiagnosability can be separated into the properties of codiagnosability together with "zero-delay codiagnosability" of "boundary safe traces". Algorithm with polynomial complexity is provided for verifying safe-codiagnosability. For a safe-codiagnosable system, the same methods as those for a codiagnosable system are applicable for the synthesis of local diagnosers as well as for on-line diagnosis using them.

REFERENCES

Boel, R. K. and J. H. van Schuppen (2002). Decentralized failure diagnosis for discrete-event systems with constrained communication between diagnosers. In: *Proceedings of International Workshop on Discrete Event Systems*.

Das, S. R. and L. E. Holloway (2000). Characterizing a confidence space for discrete event timings for fault monitoring using discrete sensing and actuation signals. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans* **30**(1), 52–66.

Debouk, R., S. Lafortune and D. Teneketzis (2000). Coordinated decentralized protocols for failure diagnosis of discrete event systems. *Discrete Event Dynamical Systems: Theory and Applications* **10**, 33–79.

Jiang, S. and R. Kumar (2004). Failure diagnosis of discrete event systems with linear-time temporal logic fault specifications. *IEEE Transactions on Automatic Control* **49**(6), 934–945.

Jiang, S., R. Kumar and H. E. Garcia (2003). Diagnosis of repeated/intermittent failures in discrete event systems. *IEEE Transactions on Automatic Control* **19**(2), 310–323.

Jiang, S., Z. Huang, V. Chandra and R. Kumar (2001). A polynomial time algorithm for diagnosability of discrete event systems. *IEEE Transactions on Automatic Control* **46**(8), 1318–1321.

Lin, F. (1994). Diagnosability of discrete event systems and its applications. *Discrete Event Dynamic Systems: Theory and Applications* **4**(1), 197–212.

Pandalai, D. and L. Holloway (2000). Template languages for fault monitoring of timed discrete event processes. *IEEE Transactions on Automatic Control* **45**(5), 868–882.

Paoli, A. and S. Lafortune (2003). Safe diagnosability of discrete event systems. In: *Proceedings of IEEE Conference on Decision and Control*. Vol. 3. Hawaii, USA. pp. 2658–2664.

Qiu, W. and R. Kumar (2004). Decentralized failure diagnosis of discrete event systems. In: *Proceedings of 2004 International Workshop on Discrete Event Systems*. Reim, France.

Qiu, W., R. Kumar and S. Jiang (2004). Decidability of distributed diagnosis under unbounded-delay communication. *IEEE Transactions on Automatic Control*. Submitted.

Rudie, K. and W. M. Wonham (1992). Think globally, act locally: decentralized supervisory control. *IEEE Transactions on Automatic Control* **37**(11), 1692–1708.

Sampath, M., R. Sengupta, S. Lafortune, K. Sinaamohideen and D. Teneketzis (1995). Diagnosability of discrete event systems. *IEEE Transactions on Automatic Control* **40**(9), 1555–1575.

Sengupta, R. and S. Tripakis (2002). Decentralized diagnosis of regular language is undecidable. In: *Proceedings of IEEE Conference on Decision and Control*. Las Vegas, NV. pp. 423–428.

Yoo, T. S. and S. Lafortune (2002). Polynomial-time verification of diagnosability of partially observed discrete-event systems. *IEEE Transactions on Automatic Control* **47**(9), 1491–1495.