Technical communiqué

Decentralized fault prognosis of discrete event systems with guaranteed performance bound

Xiang Yin\textsuperscript{a,1}, Zhaojian Li\textsuperscript{b}

\textsuperscript{a} Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA
\textsuperscript{b} Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109, USA

Abstract

We study the problem of decentralized fault prognosis of partially-observed discrete event systems. In order to capture the prognostic performance issue in the prognosis problem, we propose two new criteria: (1) all faults can be predicted $K$ steps ahead; and (2) a fault will occur for sure within $M$ steps once a fault alarm is issued; and we refer to $(M, K)$ as the performance bound of the prognostic system. A necessary and sufficient condition for the existence of a decentralized supervisor satisfying these two criteria is provided, which is termed as $(M, K)$-coprognostability. A polynomial-time algorithm for the verification of $(M, K)$-coprognostability is also proposed. Finally, we show that the proposed approach is applicable to both disjunctive and conjunctive architectures. Our results generalize previous work on decentralized fault prognosis.

1. Introduction

Fault prognosis is an important issue for safety-critical systems. Recently, the problem of fault prognosis has received considerable attention in the Discrete-Event System (DES) literature; see, e.g., Cassez and Grastien (2013), Chang, Dong, Ji, and Tong (2013), Chen and Kumar (2015), Genc and Laafortune (2009), Jéron, Marchand, Genc, and Laafortune (2008), Khoumsi and Chakib (2009), Khoumsi and Chakib (2012), Kumar and Takai (2010), Lefebvre (2014a), Lefebvre (2014b), Nouioua, Dague, and Ye (2014), Takai and Kumar (2011), Takai (2015), Ye, Dague, and Nouioua (2013), and Yin and Laafortune (2015). In Jéron et al. (2008) and Genc and Laafortune (2009), the fault prognosis (or prediction) problem was first studied for the centralized partially-observed DES, where the notion of predictability was introduced. In Kumar and Takai (2010), the authors studied the decentralized fault prognosis problem under the disjunctive architecture, where the notion of coprognostability was proposed. Particularly, a system is coprognostasable if and only if there exists a decentralized prognoser that can predict fault correctly. The decentralized prognosis problem has been further studied recently under the conjunctive architecture (Khoumsi & Chakib, 2012) and the inference-based architecture (Takai & Kumar, 2011). Roughly speaking, in the disjunctive architecture, a global fault alarm is issued iff one local agent issues a fault alarm. While in the conjunctive architecture, a global fault alarm is issued iff all local agents issue a fault alarm. In the inference architecture, multilevel inference for each local agent is used in order to issue a global fault alarm.

Most of the previous work on decentralized prognosis are based on two criteria: “no missed alarm” and “no false alarm”, where the former requires that any fault can be predicted prior to its occurrence and the latter requires that a fault will happen for sure once an alarm is issued. However, these two criteria do not care how early or how late the fault alarm is issued. In practice, once a fault alarm is issued, some procedures will be taken in order to protect the system. Since the protection could be costly, one may not want to take it unless it is necessary. On the other hand, the protection may require certain amount of time to set up. Therefore, we also need to guarantee that the fault alarm can be issued in time before certain threshold.

In this note, we investigate the problem of decentralized fault prognosis of DES. Two new prognostic performance criteria are proposed in order to capture the “timing” issue. Specifically, we require that (1) any fault can be predicted $K$ steps prior to its occurrence; and (2) if an alarm is issued, then a fault will occur for sure within $M$ steps from the alarm. We refer to this integer pair $(M, K)$ as the performance bound of the
prognostic system. The contributions of this note are as follows. First, we extend the previous work on the decentralized prognostic problem by proposing the notion of \((M, K)\)-performance bound that takes the prognostic performance issue into account. Second, we provide the necessary and sufficient condition for the existence of a decentralized prognoser that achieves this performance bound. A polynomial-time algorithm for the verification of the existence condition is also provided. Third, we show that the proposed approach is applicable to both disjunctive and conjunctive architectures. We note that the conjunctive prognosis was initially studied by Khounsri and Chakib (2012); however, to the best of knowledge, no verification algorithm for conjunctive coprognostability is provided so far. As a special case of our notion, now it can be effectively verified by the algorithm proposed in this note.

2. Preliminaries

Let \(\Sigma\) be a finite set of events and \(\Sigma^*\) be the set of all finite strings over \(\Sigma\), including the empty string \(\epsilon\). A language \(L \subseteq \Sigma^*\) is a set of strings. We denote by \(L\) the prefix-closure of \(L\), i.e., \(L = \{s \in \Sigma^* : \exists \tilde{s} \in \Sigma^* \; s.t. \; st \in L\}\). We denote by \(|s|\) the length of a string \(s \in \Sigma^*\) with \(|\epsilon| = 0\). We denote by \(L/s\) the post-language of \(s\), i.e., \(L/s = \{t \in \Sigma^* : st \in L\}\). A language \(L\) is live if \(\forall s \in L, \exists \sigma \in \Sigma : s\sigma \in L\). A DES is modeled by a deterministic finite-state automaton (DFA) \(G = (Q, \Sigma, \delta, q_0, Q_a)\), where \(Q\) is the finite set of states, \(\Sigma\) is the finite set of events, \(\delta : Q \times \Sigma \rightarrow Q\) is the partial transition function, \(q_0 \in Q\) is the initial state and \(Q_a\) is the set of marked states. We write a DFA \(G\) as \(G = (Q, \Sigma, \delta, q_0)\) if marking is not considered. The transition function \(\delta\) is extended to \(Q \times \Sigma^*\) in the usual manner (see, e.g., Cassandras & Lafortune, 2008). The language generated by \(G\) from state \(q\) is defined by \(L(G, q) = \{s \in \Sigma^* : \delta(q, s)\}\), where \(|!\) means “is defined”. The language marked by \(G\) from state \(q\) is \(L_m(G, q) = \{s \in \Sigma^* : \delta(s, q) = q_0\}\). We write \(L(G, q)\) and \(L_m(G, q)\) as \(L(G)\) and \(L_m(G)\), respectively, when \(q = q_0\). Hereafter, we assume w.l.o.g. that \(L(G)\) is live.

In the fault prognosis problem, the goal is to predict whether or not the system will violate some normal behaviors in the future. To this end, we define \(H = (Q_H, \Sigma, \delta_H, q_{0_H}, Q_{a_H})\) as the specification automaton that captures the normal behaviors of the system, where \(L(H) \subseteq L(G)\). We say that \(H\) is a sub-automaton of \(G\), denoted by \(H \subseteq G\), if \(\delta_H(q_{0_H}, s) = \delta(q_0, s)\) for all \(s \in L(G)\). We say that \(H\) is a strict sub-automaton of \(G\), denoted by \(H \subset G\), if: (1) \(H \subseteq G\); and (2) \(\forall s \in L(G) \setminus L(H) : \delta(q_0, s) \notin Q_H\). Hereafter, we assume w.l.o.g. that the specification automaton \(H\) is \((Q_H, \Sigma, \delta_H, q_{0_H})\) is a strict sub-automaton of the system automaton \(G = (Q, \Sigma, \delta, q_0)\), i.e., \(H \subset G\). Under this assumption, string \(s \in L(G)\) is a non-fault string if and only if \(\delta(s) \notin Q_H\).

In the decentralized fault prognosis (Kumar & Takai, 2010), the system is monitored by a set of agents (or local prognosers) that work as a team in order to predict the fault. We assume that there are \(n\) local agents and we denote by \(I = \{1, \ldots, n\}\) the index set. We denote by \(\Sigma_{s,i}\) the set of locally observable events of agent \(i \in I\). Then \(P_i : \Sigma^* \rightarrow \Sigma_{s,i}^*\) is the natural projection defined in the usual manner; see, e.g., (Cassandras & Lafortune, 2008). Each local prognoser \(i \in I\) is defined as the function \(A_i : P_i(L(H)) \rightarrow \{0, 1\}\), where “1” means a fault alarm is issued and “0” means no fault alarm is issued. Each local prognoser sends its local prognostic decision to a coordinator in order to calculate a global prognostic decision. The decentralized prognoser is the function \(\{A_i\}_{i \in I} : L(H) \rightarrow \{0, 1\}\) defined by: for any string \(s \in L(H)\),

\[
\{A_i\}_{i \in I}(s) = 1 \Leftrightarrow \exists i \in I : A_i(P_i(s)) = 1.
\]

In Kumar and Takai (2010), two criteria, “no missed alarm” and “no false alarm”, were proposed in order to evaluate a decentralized prognoser. In particular, it was shown that the notion of coprognostability provides the necessary and sufficient condition under which there exists a decentralized prognoser satisfying the above two conditions. We first recall its definition from Kumar and Takai (2010).

**Definition 1** (Coprognostability). A specification \(L(H)\) is said to be coprogicable w.r.t. \(L(G)\) and \(\Sigma_{s,i}\), \(i \in I\) if \(\exists m \in \mathbb{N}(\forall s \in L(G) \setminus L(H)) |\{s \in I_1 \cap L(H)\}(\forall u \in P_i^{-1}(P_i(t) \cap L(H))) \forall v \in L(G)/u| |v| \geq m \Rightarrow vu \in L(G) \setminus L(H)\).

**Remark 1.** Intuitively, coprognostability requires that for any fault string, it must have a non-fault prefix such that at least one agent knows for sure that the fault is inevitable in the future. Although the notion of coprognostability guarantees that the fault can be predicted correctly, it does not care how early or how late the fault alarm is issued, i.e., no prognostic performance is guaranteed. However, this issue is very important in many practical applications. For example, in an uninterruptible power system, one may need to predict potential failures and to take some protections before the failure occurs, e.g., starting a backup battery. On the one hand, one may not want that the fault alarm is issued too late, since the backup battery may require several steps to set up. On the other hand, one also does not want that the fault alarm is issued too early, i.e., the backup battery can only support for a limited amount of steps. Therefore, new criteria are needed in order to address the above prognostic performance requirements.

3. Main results

In this section, we propose the notion of \((M, K)\)-coprognostability that quantitatively generalizes the notion of coprognostability by taking the prognostic performance issue into account. First, we define the notion of performance bound.

**Definition 2.** Let \(M, K \in \mathbb{N}\) be two non-negative integers. A decentralized prognoser \(\{A_i\}_{i \in I}\) is said to be progicable with performance bound \((M, K)\) (or a \((M, K)\)-prognoser) if the following two properties hold:

1. Any fault can be alarmed \(K\) steps before its occurrence, i.e.,
\[
(\forall s \in L(G) \setminus L(H)) |\{i \in I_1 \cap L(H)\}(\forall u \in L(G)/s)| |v| \geq K \Rightarrow \exists i \in I_1 : A_i(P_i(s)) = 1.
\]

2. Fault is guaranteed to occur within \(M\) steps once a fault alarm is issued, i.e., for any string \(s \in L(H)\),
\[
([A_i]_{i \in I}(s) = 1) \Rightarrow (\forall i \in I_1 : s \notin L(G)/s| |v| \geq M \Rightarrow st \in L(G) \setminus L(H)).
\]

**Remark 2.** The conditions in Eqs. (2) and (3) generalize the criteria of “no missed alarm” and “no false alarm”, respectively, in a quantitative manner by requiring that when the fault alarm is issued. Note that these two performance criteria are defined in terms of event steps, i.e., we consider logical prognostic performance criteria.

Before we show the existence condition of a \((M, K)\)-decentralized prognoser, let us first introduce some necessary notations. For each state \(q \in Q_H\) in \(H\), we denote by \(d_{\min}(q)\) the length of the shortest no-fault string from \(q\) from which a fault may occur, i.e., \(d_{\min}(q) = \min_{l \in L(G)/q}(L(H)/q)| |s| - 1\). We assume w.l.o.g. that \(d_{\min}(q) \geq K\); otherwise, \((M, K)\)-coprognostability is violated trivially. Also, we denote by \(d_{\max}(q)\) the length of the longest no-fault string from \(q\), i.e., \(d_{\max}(q) = \max_{l \in L(G)/q}| |s|\). Clearly, \(d_{\max}(q) = \infty\) iff \(q\) can reach a cycle of \(H\), i.e., there exists an arbitrarily long no-fault string defined at \(q\). We denote by \(d_H(q, G)\) the set of states in
The following results reveal that therefore, it can make a local prognostic decision which is needed. 

\[ \Delta_m^i(H, G) = \{ q \in Q_i : d_{max}(q) \leq M \} \]  

For each local prognosable \( i \in I \), we also denote by \( \delta^i_\ell(s) \) Agent \( i \)'s state estimate of \( s \) under \( \Sigma_{o,i} \) w.r.t. the state space of \( H \), i.e., 

\[ \delta^i_\ell(s) := \{ q \in Q_i : \exists t \in (L(H) \cap \delta^i_\ell(t)) \} \]  

Intuitively, \( \delta^i_\ell(s) \) represents Agent \( i \)'s knowledge about the system state upon the occurrence of \( s \).

With the notions introduced above, we are now ready to introduce the notion of \((M, K)\)-coprognosability.

**Definition 3.** \( L(H) \) is said to be \((M, K)\)-coprognoasable w.r.t. \( L(G) \) and \( \Sigma_{o,i} \), \( i \in I \) if for any string \( s \in L(H) \), we have 

\[ \Delta_H^i \subseteq \delta^i_\ell(s) \subseteq \Delta_m^i(H, G). \]  

Intuitively, the above definition requires that for any string that ends up with a state in \( \Delta_m^i(H, G) \), there must exist at least one agent such that it knows for sure that the current state is in \( \Delta_m^i(H, G) \). Therefore, it can make a local diagnostic decision which is needed. The following result reveals that \((M, K)\)-coprognosability provides the necessary and sufficient condition for the existence of a \((M, K)\)-decentralized prognosor.

**Theorem 1.** There exists a \((M, K)\)-decentralized prognosor \( \{ A_i \}_{i \in I} \), if and only if, \( L(H) \) is \((M, K)\)-coprognoasable w.r.t. \( L(G) \) and \( \Sigma_{o,i} \), \( i \in I \).

**Proof.** \((\Rightarrow)\) By construction. Let us consider a decentralized prognosor \( \{ A_i \}_{i \in I} \) defined as follows. For each \( i \in I \), we define 

\[ A_i(P_i(s)) = 1 \iff \delta^i_\ell(s) \subseteq \Delta_m^i(H, G). \]  

Next, we show that the properties in Eqs. (2) and (3) are satisfied under the above prognostic strategy. First, for any fault string \( s \in L(G) \setminus L(H) \), there exists a prefix \( t \in [s] : t \in \delta_\ell(t) \). Since \( L(H) \) is \((M, K)\)-coprognoasable, we know that \( [s] = [t] \in \Delta_m^i(H, G). \) Therefore, \( \exists i \in I \) \( A_i(P_i(s)) = 1 \) i.e., \( \delta_\ell(t) = 1 \). Since \( s \in \Delta_m^i(H, G) \), we know that any \( M \)-step extension of \( t \) is still in \( L(H) \), which means that \( [s] \) is extended by \( \delta_\ell(t) \), \( \exists i \in I \) \( [s] = [t] \in \Delta_m^i(H, G). \) Therefore, Eq. (2) holds. Second, we show Eq. (3) holds by contradiction. We assume that Eq. (3) does not hold, which implies that \( \exists s \in L(G) : [s] = 1 \). However, by \( \{ A_i \}_{i \in I} \) defined as follows, we have that \( [s] = [t] = 1 \) for all strings \( s \) such that \( (s) \subseteq \Delta_m^i(H, G) \). Since \( P_i(s) = P(s) \), we know that \( v(t) \in [s] : P(t) = P(t) \). Moreover, since \( \delta_\ell(t) \notin \Delta_m^i(H, G) \), we know that \( v(t) \notin [s] : P(t) = P(t) \). Therefore, Eq. (2) holds. Thus, Eq. (3) also holds.

\((\Leftarrow)\) By contradiction. We assume that there exists a decentralized prognosor \( \{ A_i \}_{i \in I} \) but \( L(H) \) is not \((M, K)\)-coprognoasable, which means that \( \exists s \in L(G) : [s] = 1 \). However, by \( \{ A_i \}_{i \in I} \) defined as follows, we have that \( [s] = [t] = 1 \) for all strings \( s \) such that \( (s) \subseteq \Delta_m^i(H, G) \). Since \( P_i(s) = P(s) \), we know that \( v(t) \in [s] : P(t) = P(t) \). Moreover, since \( \delta_\ell(t) \notin \Delta_m^i(H, G) \), we know that \( v(t) \notin [s] : P(t) = P(t) \). Therefore, Eq. (2) holds. Thus, Eq. (3) also holds. \( \Box \)
Proof. \(\Rightarrow\) By contrapositive. Suppose that \(L_m(V) \neq \emptyset\), i.e., \(\exists s = (s_1, s_2, s_3) \in L(V)\) such that \(f_{\delta_2}(q_{0V}, s) \in \Delta_m\), i.e., \(\delta_0(s_1) \in \delta_{O}(H, G)\) and \(\delta_2(s_2), \delta_0(s_3) \notin \Xi_m(H, G)\). By the construction of \(V\), we have \(P_1(s_1) = P_2(s_2)\). To see this, assume that \(P_1(s_1) \neq P_2(s_2)\). Then, we know that \(\exists (s_1', s_2', s_3') \in \Sigma^3 \) where \(\delta_1(s_1') \neq \delta_0(s_2')\) and \(\delta_0(s_3') \notin \Xi_m(H, G)\). By the construction of \(V\), we see that \((s_1, s_2, s_3)\) is defined at \(f_{\delta_2}(q_{0V}, (s_1', s_2', s_3'))\) only if \(\delta_0(s_1) = P_2(s_2)\). Therefore, \(P_1(s_1) \neq P_2(s_2)\) is not possible. Moreover, \(P_1(s_1) = P_2(s_2)\) implies \(\delta_0(s_1), \delta_0(s_3) \subseteq \Xi_m(H, G)\). However, since \(\delta_0(s_3) \notin \Xi_m(H, G)\), we have that \(\delta_0(s_1) \notin \Xi_m(H, G)\). Similarly, we have that \(\delta_0(s_1) \notin \Xi_m(H, G)\). Therefore, \(L(H)\) is not \((M, K)\)-coprognosable.

\(\Leftarrow\) By contrapositive. Suppose that \(L(H)\) is not \((M, K)\)-coprognosable. Then \(\exists s \in L(H) : \delta_0(s) \in \delta_{O}(H, G)\) and \(\delta_0(s) \in I\). Since \(\delta_0(s) \notin \Xi_m(H, G)\), we know that there must exist a string \(s_1\) such that \(P_1(s_1) = P_2(s_1)\) and \(\delta_0(s_1) \notin \Xi_m(H, G)\). Similarly, for Agent 2, there also exists a string \(s_1\) such that \(P_2(s_1) = P_2(s_2)\) and \(\delta_0(s_2) \notin \Xi_m(H, G)\). Since \(P_1(s_1) = P_2(s_2)\) and \(P_2(s_2) = P_2(s_2)\), by the construction of \(V\), we know that \(\delta_0(s_1), \delta_0(s_2) \subseteq \Xi_m(H, G)\). Therefore, \(\delta_0(s_1) \notin \Xi_m(H, G)\), \(\delta_0(s_2) \notin \Xi_m(H, G)\), and \(\delta_0(s_3) \notin \Xi_m(H, G)\). Therefore, \(L_m(V) \neq \emptyset\).

**Example 1.** Consider the system in Fig. 1(a), where state \(F\) denotes the single fault state, i.e., the specification automaton \(H\) is obtained by removing \(F\) from \(G\) and taking the accessible part. We assume that \(I = \{1, 2\}\) and let \(\Sigma_{1} = \{a, 0\}\) and \(\Sigma_{2} = \{b, 0\}\), i.e., \(e, f\) and \(f\) are both globally unobservable. We consider \(M = 4\), \(K = 2\) and we want to verify whether or not the specification is \((4, 2)\)-coprognosable. Part of the verifier of this system is shown in Fig. 1(b). We see that state \((6, 2, 5)\) is reached. However, for state \(6\), we know that \(d_m(6) = 2\), i.e., \(6 \in \delta_{O}(H, G)\). Moreover, since both states 2 and 5 are in a cycle of \(H\), we know that \(d_m(2) = d_m(5) = \infty\), i.e., 2, 5 \(\notin \Xi_m(H, G)\). Therefore, \((6, 2, 5)\) is a marked state in \(V\) and this system is not \((4, 2)\)-coprognosable.

We conclude this section by discussing the complexity of the above verification procedure. For each state \(q \in Q_m\), in order to compute \(d_m(q)\), we just need to find the shortest path from \(q\) to a state in \(Q \setminus Q_m\), which can be done in \(O(|Q_m| \Sigma)\); see, e.g., (Sedgewick & Wayne, 2011) (p. 652). To compute \(d_m(q)\), first, we need to compute all strongly connected components, i.e., cycles, which can be done by Kosaraju’s algorithm in \(O(|Q_m| \Sigma)\); see, e.g., (Sedgewick & Wayne, 2011) (p. 587). Then, for any state \(q\) in a cycle, we assign \(d_m(q) = \infty\). Next, we take a backwards search from states with infinite value in \(G\), and assign each state reachable in the backwards search an infinite value, since this means that this state can reach a cycle. This step is just depth-first search, which is still in \(O(|Q_m| \Sigma)\). Finally, we remove all states with infinite value from \(H\) and the resulting system is acyclic, since all states in cycles have been removed. Then we used the longest path search algorithm for the acyclic case provided in Sedgewick and Wayne (2011) (p. 661) to determine the values of states remained. The complexity of this step is still \(O(|Q_m| \Sigma)\). In the worst case, the \((M, K)\)-verifier has \(|Q_m|^3\) states and \(3|\Sigma|^2\) transitions. Therefore, constructing \(V\) can be done in \(O(|\Sigma|^2|Q_m|^3)\).

**4. Fault prognosis using conjunctive architecture.**

In the above development, the global decision of the decentralized prognoser is “1” iff there exists one local agent whose local decision is “1”. This decentralized information structure is usually referred to as the *disjunctive architecture*. However, one can also use the *conjunctive architecture* in order to fuse the local decisions. Specifically, under the conjunctive architecture, the decentralized prognoser \(A_l\) is defined by: for any string \(s \in L(H)\:

\[
A_l(s) = 1 \iff \forall i \in I: A_i(P_i(s)) = 1.
\]

(9)

Hereafter, we refer to \((M, K)\)-coprognosability defined in definition (6) as \((M, K)\)-*disjunctive* coprognosability. We show how the results developed for the disjunctive architecture can be extended to the conjunctive architecture.

**Definition 4.** \(L(H)\) is said to be \((M, K)\)-*conjunctively coprognosable* w.r.t. \(L(G)\) and \(\Delta_{m+1}, i \in I\) if for any string \(s \in L(H)\),

\[
\delta_0(s) \notin \Xi_m(H, G) \Rightarrow (\exists i \in I) (\delta_0(s) \cap \delta_0(H, G) = \emptyset).
\]

(10)

Similar to the disjunctive case, the above definition requires that for any string that ends up with a state that is not in \(\Xi_m(H, G)\), there must exist one agent that knows for sure that the current state is not in \(\delta_0(H, G)\). Therefore, it will not make a wrong local prognostic decision. The following result reveals that \((M, K)\)-conjunctive coprognosability provides the necessary and sufficient condition for the existence of a \((M, K)\)-decentralized prognoser under the conjunctive architecture.

**Theorem 3.** Under the conjunctive architecture, there exists a decentralized prognoser \(A_l\) satisfying Eqs. (2) and (3), if and only if, \(L(H)\) is \((M, K)\)-conjunctively coprognosable w.r.t. \(L(G)\) and \(\Delta_{m+1}, i \in I\).

**Proof.** \(\Leftarrow\) By construction. Let us consider a decentralized prognoser \(A_l\) as follows. For each \(i \in I\), we define

\[
A_i(P_i(s)) = 1 \iff \delta_0(s) \cap \delta_0(H, G) = \emptyset.
\]

(11)

First, for any string \(s \in \delta_0(H, G)\), we have \(\delta_0(s) \notin \delta_0(H, G)\), \(\forall i \in I\). Therefore, \(\exists i \in I: (\delta_0(s) \cap \delta_0(H, G) = \emptyset)\). Second, we show Eq. (3) holds by contradiction. We assume that Eq. (3) does not hold, which implies that \(\exists s \in L(H) : \{A_l(s) = 1\} \cap \{i \in I: A_i(s) = 1\} \neq \emptyset\). Then, for the above strings \(s\), we know that \(\forall i \in I: \exists s \in L(H) : (P_i(s) = P_i(s) \cap \delta_0(s) \in \delta_0(H, G))\). By contradiction, we know that \(\exists i \in I: (\delta_0(s) \cap \delta_0(H, G) = \emptyset)\), which means that at least one agent’s decision is “0”. Therefore, \(\forall i \in I\), \(A_l(s) = 0\), which is a contradiction.

**Remark 5.** To verify \((M, K)\)-conjunctive coprognosability, we can still use the \((M, K)\)-verifier proposed in the previous section. In this case, instead of defining the set of marked states \(X_m\) according to Eq. (8), one can define \(X_m\) by

\[
X_m := \{(q_1, q_2, q_3) \in \Delta_m: q_1 \notin \Xi_m(H, G) \& q_2, q_3 \in \delta_0(H, G)\}.
\]

It is easy to verify that \(L(H)\) is \((M, K)\)-conjunctively coprognosable iff \(L_m(V) = \emptyset\) for the modified \(V\). The proof is omitted here since it is similar to the proof of Theorem 2.
The above results generalize the conjunctive prognosis studied in Khoumsi and Chakib (2012) in twofold. First, in Khoumsi and Chakib (2012), only the lower bound $K$ is guaranteed for the performance of the prognostic system while our results guarantee both lower bound $K$ and upper bound $M$. Second, although the notion of conjunctive coprognosability was introduced in Khoumsi and Chakib (2012), there is no verification algorithm provided so far in the literature. Clearly, as a special case of $(M,K)$-conjunctive coprognosability, conjunctive coprognosability can also be effectively verified by using the $(M,K)$-verifier as we discussed in the above remark.

Remark 6. In fact, one can verify that the system in Fig. 1, which is not $(4,2)$-disjunctively coprognosable, is $(4,2)$-conjunctively coprognosable. It is also not difficult to find a system that is not $(M,K)$-conjunctively coprognosable but $(M,K)$-disjunctively coprognosable; for example for this is provided in Fig. 2. Therefore, $(M,K)$-disjunctive coprognosability and $(M,K)$-conjunctive coprognosability are incomparable. By comparing Eqs. (7) and (11), we see that two different prognostic strategies are used for these two different architectures, respectively. For the disjunctive case, the strategy we take is “alarm as early as possible”, since according to Eq. (7), a fault alarm is issued immediately when one agent knows for sure that the fault is inevitable within $M$ steps. However, for the conjunctive case, the strategy we take is “alarm as late as possible”, since according to Eq. (11), such a fault alarm will not be issued until a state in $\mathcal{O}_k(H, G)$ is reached. Clearly, we cannot postpone the alarm any more; otherwise, the condition in Eq. (2) will be violated.

5. Conclusion

In this note, we extended previous work on the problem of decentralized fault prognosis by taking the prognostic performance issue into account. The notion of $(M,K)$-coprognosability was introduced as the necessary and sufficient condition for the existence of a decentralized prognoser satisfying certain time constraints. A polynomial-time algorithm for the verification of $(M,K)$-coprognosability was provided. We showed that the proposed approach is also applicable to the conjunctive architecture for which no verification algorithm is provided so far. Note that the prognostic performance criteria considered in the paper are logical; investigating numerical performance criteria for real-time DES is also an interesting future direction.

Acknowledgments

The authors would like to thank the anonymous reviewers for their useful comments on improving this paper.

References