An Optimized Algorithm for Diagnosability of Component-based Systems

Lina Ye, Philippe Dague

An Optimized Algorithm for Diagnosability of Component-based Systems

Lina.YE * **Philippe.DAGUE** *

* *Univ Paris-Sud, LRI; CNRS, Orsay; INRIA Saclay–Ile-de-France, Orsay, 4 rue Jacques Monod, bat G, Orsay, F-91893, France (e-mail: lina.ye@lri.fr, philippe.dague@lri.fr)*

Abstract: Diagnosability is a crucial system property that determines at design stage how accurate any diagnosis algorithm can be on a partially observable system. The existence of two indistinguishable behaviors, i.e. holding the same observations, with exactly one of them containing the fault violates the diagnosability property. A classical approach for diagnosability verification consists in constructing a finite state machine called twin plant to search for a path representing such indistinguishable behaviors, called a critical path. To avoid the unrealistic hypothesis about the monolithic model of a complex system, recent work constructs local twin plants and then incrementally synchronizes some of them until diagnosability is decided without computing the impractical global twin plant. In this paper, we optimize the distributed approach by abstracting necessary and sufficient diagnosability information from local twin plants to check the existence of critical paths. Thus diagnosability can be analyzed with as small search space as possible. Furthermore, our approach describes how to improve the diagnosis algorithm by using our diagnosability results in a formal way when the system is verified to be diagnosable. Finally, when the system is not diagnosable, the algorithm returns some useful information about its indistinguishable behaviors, which can help in upgrading system diagnosable level.

Keywords: discrete event systems, fault diagnosis, models, algorithms, distributed systems

1. INTRODUCTION

Automated fault diagnosis has a significant economic impact on the improvement of performance and reliability of complex discrete event systems (DESs). This problem has received considerable attention from Artificial Intelligence community and Control community. Generally speaking, diagnosis reasoning is to detect possible faults that can explain the observations continuously received by a monitor from a system. However, the accuracy of diagnosis depends on diagnosability of the system. Diagnosability is an important system property that determines at design stage how accurate any diagnosis algorithm can be on a partially observable system. The diagnosability results can either help in choosing the diagnosis algorithm when system is diagnosable or provide some information about how to redesign the system to improve its diagnosable level when it is not diagnosable.

The existence of two indistinguishable behaviors, i.e. holding the same observations, with exactly one of them containing the fault violates diagnosability property. The classical and centralized diagnosability checking methods are to check the existence of such indistinguishable behaviors with the assumption that the knowledge about the system is a monolithic model. This hypothesis is normally unrealistic when dealing with real complex systems due to the combinatorial explosion of the search space. So we propose here a formal framework for checking diagnosability of component-based systems, where the problem is described as a distributed search problem to avoid calculating global objects.

In this paper, our proposed approach makes several contributions to the diagnosability problem. First, we gear classical diagnosability definition for an entire system to regional diagnosability one for a subsystem, which leads to defining a diagnosable subsystem. And we describe how to improve diagnosis algorithm in terms of observation reduction with a given diagnosable subsystem in a formal way. Second, we provide a new distributed theoretical framework to check regional diagnosability and thus diagnosability of component-based systems. Instead of performing diagnosability verification on the global twin plant or local twin plants, we abstract necessary and sufficient diagnosability information from local twin plants and then distribute the search on these abstracted ones. This algorithm is optimized in the sense that with our abstracted diagnosability information, the search state space is reduced to be as small as possible. Third, the diagnosability results we obtain can help in improving the diagnosis algorithm when the system is diagnosable, in which case the algorithm returns a diagnosable subsystem. Otherwise, the algorithm provides some helpful information about indistinguishable behaviors that can be used to upgrade the diagnosable level of the system when the system is verified to be not diagnosable.

The paper is organized as follows. In next section we model a component-based system as a set of finite state machines (FSMs), review classical diagnosability definition and twin plant method. Then section 3 defines regional diagnosability before describing how to abstract necessary and sufficient diagnosability information from local twin plants and how to distribute diagnosability verification on these abstracted ones. And then the formal algorithm is given in section 4 before section 5 discusses its results. Finally, some related work is presented in section 6 before the conclusion.

2. BACKGROUND

Now we review the classical diagnosability definition of DESs and then recall twin plant method for diagnosability checking.

2.1 Diagnosability of Discrete Event Systems

We consider a distributed DES composed of a set of components $G_1, ..., G_n$ that can communicate with each other by communication events. Such a system is modeled by a set of FSMs, each one modeling a component (Pencolé (2004)).

Definition 1. **(Local Model)** A component G_i is modeled as a FSM, denoted by $G_i = (Q_i, \Sigma_i, \delta_i, q_i^0)$, where

- Q_i is the set of states
- Σ_i is the set of events
- $\delta_i \subseteq Q_i \times \Sigma_i \times Q_i$ is the set of transitions
- q_i^0 is the initial state

The set of events Σ_i is divided into four disjoint parts: Σ_{i_o}, the set of observable events, Σ_{i_f}, the set of unobservable fault events, Σ_{i_u}, the set of unobservable normal events and Σ_{i_c}, the set of unobservable communication events that are shared by at least two components. The only shared events between different components are communication events. For the transition set, it is easy to extend $\delta_i \subseteq Q_i \times \Sigma_i \times Q_i$ to $\delta_i \subseteq Q_i \times \Sigma_i^* \times Q_i$ by the following way: 1) $(q, \epsilon, q) \in \delta_i$, where ϵ is the null event; 2) $(q, se, q1) \in \delta_i$ if $\exists q\prime \in Q, (q, s, q\prime) \in \delta_i$, $(q\prime, e, q1) \in \delta_i$, where $s \in \Sigma_i^*, e \in \Sigma_i$.

The global model of the entire system is implicitly defined as the synchronized product of all component models based on their shared events, here communication events, denoted by $G = Sync(G_1, ..., G_n)$. In the synchronization, any shared event always occurs simultaneously in all components that define it and the result is a FSM whose state space is the Cartesian product of the state spaces of the components. Figure 1 depicts a distributed system composed of three components G_1 (top), G_2 (bottom left) and G_3 (bottom right).

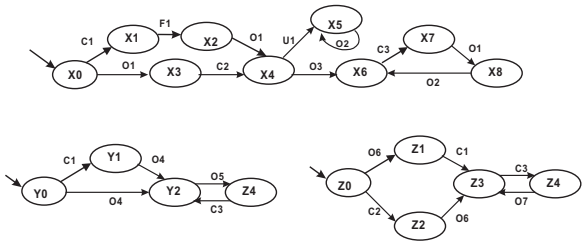

Fig. 1. A system with three components G_1 (top), G_2 (bottom left) and G_3 (bottom right). The events Oi denote observable events, the events Fi denote unobservable fault events, the events Ui denote unobservable normal events and the events Ci denote the unobservable communication events.

Let $G = (Q, \Sigma, \delta, q^0)$ be the global model of a system, then the prefix-closed language $L(G)$ generated by G describes the behaviors of this system, where $L(G) \subseteq \Sigma^*$. Formally, the language $L(G)$ is the set of words produced from G: $L(G) = \{s \in \Sigma^* | \exists q \in Q, (q^0, s, q) \in \delta\}$. In the following, we call words from G as trajectories in G. A path in G is a sequence $q_0\sigma_0q_1\sigma_1...$, where $\sigma_0\sigma_1...$ is a trajectory in G and $\forall i$, we have $(q_i, \sigma_i, q_{i+1}) \in \delta$. The synchronized FSM on any non-empty set $\{G_{i_1}, ...G_{i_m}\}$ is called a subsystem of the system G, denoted by G_S, where $G_{i_k}, k \in \{1, ...m\}$ could be any component in the system. We define projection P and reverse projection P^{-1} with respect to two event sets Σ and $\Sigma\prime$, where $\Sigma\prime \subseteq \Sigma$, as follows:

- $P_{\Sigma \to \Sigma\prime}(\epsilon) = \epsilon$
- $\forall \sigma \in \Sigma$, $P_{\Sigma \to \Sigma\prime}(\sigma) = \sigma$, when $\sigma \in \Sigma\prime$
- $\forall \sigma \in \Sigma$, $P_{\Sigma \to \Sigma\prime}(\sigma) = \epsilon$, when $\sigma \in \Sigma \backslash \Sigma\prime$
- $\forall s \in \Sigma^*, \forall \sigma \in \Sigma$, $P_{\Sigma \to \Sigma\prime}(s\sigma) = P_{\Sigma \to \Sigma\prime}(s)P_{\Sigma \to \Sigma\prime}(\sigma)$
- $\forall s \in \Sigma\prime^*$, $P^{-1}_{\Sigma\prime \to \Sigma}(s) = \{t \in \Sigma^* | \ P_{\Sigma \to \Sigma\prime}(t) = s\}$

In this paper, we have the following assumption.

Assumption 1. Both the language and the observable language of any component are alive.

The assumption means that in any component, there is a transition from every state and the observable projection of any infinite local trajectory ($P_{\Sigma_i \to \Sigma_{i_o}}(p)$) is infinite. In other words, there is no cycle with only unobservable events in any component.

We now review the diagnosability definition for DESs introduced in Sampath et al. (1995). A system is diagnosable with respect to a fault f iff for any trajectory ending with the occurrence of f, denoted by s^f, for any extension t of s^f with a sufficiently large finite number of events, any trajectory with the same observations as $s^f.t$ also contains the occurrence of f. This is formally defined as:

Definition 2. **(Diagnosability)** A fault f is diagnosable in the system G, iff

$$\forall s^f t, \exists d \in N, \text{if } |t| > d, \text{then}$$
$$\forall p \in P^{-1}_{\Sigma_o \to \Sigma}(P_{\Sigma \to \Sigma_o}(s^f t)), f \text{ occurs in } p.$$

If f is diagnosable in G, we call G a f-diagnosable system. In this definition, the system is assumed to have a monolithic model. The diagnosability checking consists in searching for a pair of trajectories p and $p\prime$ satisfying three conditions (Cassandras and Lafortune (2008)): 1) p contains f and $p\prime$ does not; 2) p has arbitrarily many events after f; 3) $P_{\Sigma \to \Sigma_o}(p) = P_{\Sigma \to \Sigma_o}(p\prime)$. Such two trajectories are called a **critical pair**, which witnesses non-diagnosability.

2.2 Twin Plant Method

The basic idea of a twin plant, described in Jiang et al. (2001), is to build a FSM that compares every pair of trajectories with the same observations to search for critical pairs. We first construct local diagnosers from local models, which in turn serve to compute local twin plants.

Definition 3. **(Local Diagnoser)** The local diagnoser of the component G_i is the FSM $D_i = (Q_{D_i}, \Sigma_{D_i}, \delta_{D_i}, q^0_{D_i})$ where

- $Q_{D_i} \subseteq Q_i \times F$, $F \subseteq 2^{\Sigma_{i_f}}$ is the set of states
- $\Sigma_{D_i} = \Sigma_{i_o} \cup \Sigma_{i_c}$ is the set of events
- $\delta_{D_i} \subseteq Q_{D_i} \times \Sigma_{D_i} \times Q_{D_i}$ is the set of transitions
- $q^0_{D_i} = (q_i^0, \emptyset)$ is the initial state

The transitions of δ_{D_i} are those $((q, q_f), e, (q\prime, q_f\prime))$ satisfying the following condition, with (q, q_f) reachable from the initial state $q^0_{D_i}$: there is a transition sequence $p = (q \xrightarrow{uo_1} q_1 ... \xrightarrow{uo_m} q_m \xrightarrow{e} q\prime)$ in G_i with $uo_k \in \Sigma_{i_u} \cup \Sigma_{i_f}, \forall k \in \{1, ..., m\}$, $e \in \Sigma_{i_o} \cup \Sigma_{i_c}$ and $q_f\prime = q_f \cup (\{uo_1, ...uo_m\} \cap \Sigma_{i_f})$.

Local diagnoser shows all possible faults after any sequence of observable events and communication events. The top part of figure 2 presents the local diagnoser of the component G_1. Then a local twin plant is obtained by synchronizing the local diagnoser with itself based on the observable events to obtain all pairs of local trajectories with the same observations. The two identical diagnosers are denoted by D_i^l (left instance) and D_i^r (right instance). Since this synchronization is based on the set of observable events, hence the non-synchronized communication events are distinguished between the two instances by the prefix of L and R: in D_i^l (D_i^r), each communication event $c \in \Sigma_{i_c}$ from D_i is renamed by $L : c$ ($R : c$) and all their observable events do not change their name.

Definition 4. **(Local Twin Plant)** The local twin plant of the component G_i is the FSM $T_i = Sync(D_i^l, D_i^r)$.

Each state of a local twin plant is a pair of local diagnoser states that provide two possible diagnoses with the same local observations. Given a twin plant state $((q^l, q_f^l)(q^r, q_f^r))$, if the fault $f \in q_f^l \cup q_f^r$ but $f \notin q_f^l \cap q_f^r$, which means that the occurrence of f is not certain in this state, then this twin plant state is called an ambiguous state with respect to f. An ambiguous state cycle is a cycle containing only ambiguous states. The bottom part of figure 2 depicts a part of local twin plant T_1 of the component G_1, where each state label (top) is composed of a state label of D_i^l (middle) and that of D_i^r (bottom). The gray nodes represent ambiguous states with respect to $F1$, which form ambiguous state cycles.

In the synchronized product of a set of local twin plants based on their communication events (left communication events with left ones and right communication events with right ones), $Sync(T_{s_1}, ...T_{s_m})$, where $\{s_1, ..., s_m\} \subseteq \{1, ...n\}$, any state is composed of a set of local twin plant states $q^t = (q_{s_1}^t, ..., q_{s_m}^t)$, where $q_{s_i}^t$ represents a state of T_{s_i}. If there exists an ambiguous state $q_{s_i}^t$ in q^t, then q^t is called an ambiguous state. If this set includes the local twin plants of all components, their synchronization defines the global twin plant. A path in the global twin plant is called a global critical path if it contains an ambiguous state cycle with at least one observable event, which corresponds to a critical pair. Thus diagnosability checking is to search for such a path in the global twin plant. From Jiang et al. (2001), we can obtain the following fundamental theorem.

Theorem 5. The fault f is diagnosable in a system G iff there is no global critical path.

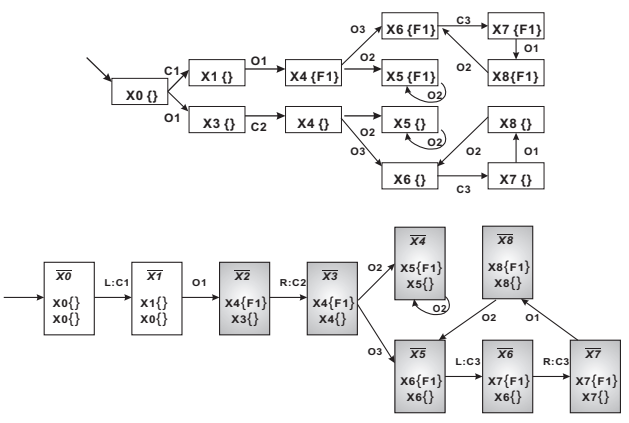

Fig. 2. Local diagnoser D_1(top) and part of local twin plant T_1 (bottom) of component G_1 (see figure 1).

3. DISTRIBUTED FRAMEWORK

The non existence of global critical paths verifies diagnosability property of a system. The distributed idea is to decide diagnosability without necessarily building the global twin plant, whose construction can be very computationally demanding for large and complex systems. In this section, we first define regional diagnosability for component-based systems and then show how to abstract necessary and sufficient diagnosability information from local twin plants and how to check regional diagnosability based on the abstracted version of local twin plants with as small search space as possible.

3.1 Regional Diagnosability

For a component-based system, definition 2 can be geared towards a subsystem G_S containing a subset of components. Let Σ_S denote the event set of the subsystem G_S and Σ_{S_o} denote the observable event set of G_S. We formally define regional diagnosability as follows:

Definition 6. **(Regional Diagnosability)** A fault f is regionally diagnosable in a system G with respect to a subsystem G_S, where $f \in \Sigma_S$, iff

$$\forall s^f t, \exists d \in N, \text{if } |P_{\Sigma \to \Sigma_{S_o}}(t)| > d, \text{ then}$$
$$\forall p \in P^{-1}_{\Sigma_{S_o} \to \Sigma}(P_{\Sigma \to \Sigma_{S_o}}(s^f t)),\ f \text{ occurs in } p.$$

If f is regionally diagnosable in G with respect to G_S, then G is called a f^{G_S}-diagnosable system and G_S is called a diagnosable subsystem. In such a system, we are sure that f has effectively occurred on G_S when we observe enough long events from G_S after the occurrence of f. Thus the observations from G_S are sufficient for diagnosis decision with respect to the fault f, denoted by D_f. Given a f^{G_S}-diagnosable system, the diagnosis decision can be defined as follows: 1) $D_f(p) = 1$, when $\forall p\prime \in P^{-1}_{\Sigma_{S_o} \to \Sigma}(P_{\Sigma \to \Sigma_{S_o}}(p))$, f occurs in $p\prime$; 2) otherwise, $D_f(p) = 0$. Here $D_f(p)$ being 1 means the effective occurrence of f on the trajectory p, otherwise $D_f(p)$ is 0. Note that with a diagnosable subsystem G_S, only observations from G_S are involved. Otherwise, we need all observations in the whole system to decide diagnosis.

Lemma 7. If a system G is f^{G_S}-diagnosable, then it is $f^{G_{S\prime}}$-diagnosable, where $G_S \subseteq G_{S\prime}$.

Proof Suppose that G is f^{G_S}-diagnosable and not $f^{G_{S\prime}}$-diagnosable, where $G_S \subseteq G_{S\prime}$. Since G is f^{G_S}-diagnosable, from definition 6, after a finite number of observations from G_S, we are sure that f has effectively occurred after the occurrence of f. Then from the fact that the observable events between components intersect and from assumption 1 that implies no loop of unobservable events in any component, the occurrence of f can be determinable after a finite number of observations from the subsystem $G_{S\prime}$. It follows that G is $f^{G_{S\prime}}$-diagnosable, which contradicts the assumption.

Lemma 7 means that the existence of a diagnosable subsystem verifies the diagnosability property of the system. Let G be a f^{G_S}-diagnosable system. If $\forall G'_S, G'_S \subset G_S$, G is not $f^{G'_S}$-diagnosable, then G_S is called a minimal diagnosable subsystem with respect to f, which is not necessarily unique. Next we will show how to incrementally check regional diagnosability, which in turn helps to improve diagnosis algorithm as described above.

3.2 Diagnosability Information Abstraction

We now present how to abstract diagnosability information from local twin plants. In the following, let G_f denote the component where the fault f may occur. In a local twin plant, a local path containing an ambiguous state cycle is called a local possible critical path, LPCP for short. The relations between global critical paths and LPCPs can be concluded as follows: 1) if there exists a global critical path in the global twin plant, its projection on the local twin plant T_f must be a LPCP; 2) if there is no LPCP in T_f, then there is no global critical path in the global twin plant; 3) if there is a LPCP in T_f, then it may or may not be extended as a global critical path in the global twin plant when synchronizing with other local twin plants. What we are interested in is the case 3, where the diagnosability information, i.e. LPCPs, originates only in the local twin plant T_f. So our goal is to determine if the LPCPs are going to survive in the global twin plant with as small space as possible instead of computing it. Since all local paths of local twin plants are synchronized via communication events, they can only be blocked by communication events. So it suffices to consider only the LPCPs in T_f and the communication events in other relative local twin plants.

We now define the operation of Delay Communication Closure on a FSM G, denoted by $\complement(G)$, where the communication information of G is preserved. Given a FSM $G\ (Q, \Sigma, \delta, q^0)$, its Delay Communication Closure is $\complement(G) = (Q_\complement, \Sigma_\complement, \delta_\complement, q^0)$, where $Q_\complement \subseteq Q, \Sigma_\complement = \Sigma_c$, $\delta_\complement(q, \sigma) = q\prime$ if $\delta(q, s\sigma) = q\prime$ in G and $\forall \sigma\prime \in s, \sigma\prime \notin \Sigma_c$, $\sigma \in \Sigma_c$. Here Σ_c is the set of communication events in G.

Definition 8. **(Abstracted Local Twin Plant-ALTP)** The abstracted local twin plant (ALTP) from a local twin plant T_i, denoted by T_i^a, is obtained by the following steps:

(1) Delay Communication Closure is operated on the local twin plant T_i, $T_i^a = \complement(T_i)$.

(2) If there exists a local path in T_i: $q_0 \xrightarrow{e_1} q_1 ... \xrightarrow{e_n} q_n$, where q_0 is the initial state of T_i, $\exists j \in \{0, ..., n-1\}$, $q_j = q_n$, $\forall k \in \{j+1, ..., n\}, e_k \in \Sigma_{i_o}$ and all q_k are ambiguous states with respect to the set of faults, denoted by F, suppose that the corresponding local path in T_i^a is $p = q_0' \xrightarrow{c_1} q_1' ... \xrightarrow{c_m} q_m'$, then it is modified as $p\prime = q_0' \xrightarrow{c_1} q_1' ... \xrightarrow{c_m} q_m' \xrightarrow{obs_i} q^F \xrightarrow{obs_i} q^F$, where obs_i represents at least one observable event of component G_i, and q^F represents an ambiguous twin plant state with respect to the set of faults F, whose ambiguity is the same as $q_k, \forall k \in \{j+1, ..., n\}$. Note that if q^F is not ambiguous to any fault, then $F = \emptyset$.

The ALTP T_f^a keeps the corresponding part of all ambiguous state cycles of all LPCPs in T_f. Since the event set of a local twin plant is the set of communication events and observable events and there is no cycle of unobservable events in any component, an ambiguous state cycle could only be two types: 1) with both communication events and observable events; 2) with only observable events. Clause(1) of definition 8 keeps the ambiguous state cycles in T_f of the first type. For the second type, operating Delay Communication Closure on T_f of clause(1) loses those ambiguous state cycles with only observable events. So we recuperate them through clause(2) by adding their corresponding ambiguous state cycle with at least one observable event. Considering that ambiguous state cycles only originate in T_f, clause(2) is only for T_f^a construction. To construct all other ALTPs, only clause(1) is performed to abstract communication information, which is the only reason that can block LPCPs and make ambiguous state cycles disappear. In T_f^a, the corresponding local path of a LPCP from T_f is still called a LPCP since it preserves all ambiguous state cycles. The ALTP thus keeps all necessary and sufficient diagnosability information but is practically much smaller than its corresponding local twin plant. Figure 3 illustrates part of ALTPs T_1^a (top), T_2^a (bottom left) and T_3^a (bottom right) of components G_1, G_2 and G_3, respectively. Here Obs_1 represents at least one observable event of component G_1. And $q^{\{F1\}}$ represents an ambiguous twin plant state with respect to $F1$. Only T_1^a contains diagnosability information, i.e. ambiguous state cycles formed by gray nodes.

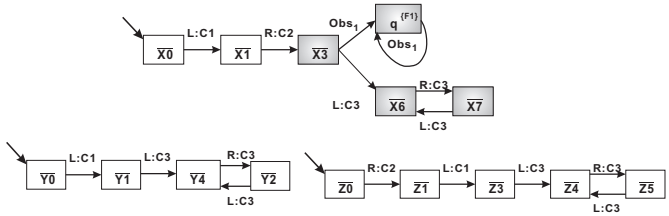

Fig. 3. Part of ALTP T_1^a(top), part of ALTP T_2^a (bottom left) and part of ALTP T_3^a (bottom right) of components G_1, G_2, G_3, respectively (see figure 1).

3.3 Distributed Verification

The reachability of LPCPs from T_f^a in the global twin plant can be determined by synchronizing T_f^a with other ALTPs. Now we first define a globally consistent LPCP. Given a local path ρ in T_f^a, if it does not disappear and contains an ambiguous state cycle in the synchronized product of the set of ALTPs, $Sync(T_f^a, T_{s_1}^a, ..., T_{s_m}^a)$, where $\{f, s_1, ...s_m\} \subseteq \{1, ..., n\}$, then ρ is a consistent LPCP of the subsystem $G_S = Sync(G_f, G_{s_1}, ..., G_{s_m})$. If $G_S = G$ or if there is no communication event in G_S that is also contained in $G \backslash G_S$, then ρ is called a *globally consistent LPCP*. In the latter case, all communication events in ρ are validated in terms of the interactions with its neighborhood. Thus it will still be a consistent LPCP when current subsystem is extended to the whole system.

Lemma 9. A local path in T_f^a is a globally consistent LPCP iff it corresponds to a global critical path.

Proof :
($\Rightarrow$) Suppose a local path ρ in T_f^a is a globally consistent LPCP and that it does not correspond to a global critical path. Recall that a global critical path is a path in the global twin plant containing an ambiguous state cycle with at least one observable event. It follows that there are three causes leading to non correspondence of ρ to a global critical path: 1) ρ disappears when synchronizing with some other ALTPs, which means that it has no corresponding path in the global twin plant; 2) ρ has a corresponding path in the global twin plant that has no ambiguous state cycle; 3) ρ has a corresponding path in the global twin plant that has ambiguous state cycles but none of them contain observable events. From the definition of a globally consistent LPCP mentioned as above, the first two cases indicate that ρ is not a globally consistent LPCP. Then assumption 1 implies that there is no cycle with only unobservable events in any component of the system, which makes case 3 impossible. Now we can deduce that ρ is not

a globally consistent LPCP, which contradicts the assumption that ρ is a globally consistent LPCP.
($\Leftarrow$) Now suppose that a local path ρ in T_f^a corresponds to a global critical path but it is not a globally consistent LPCP. Its non global consistency means that either it disappears when synchronizing with some other ALTPs or its corresponding path in the global twin plant has no ambiguous state cycle. Both cases imply that ρ has no corresponding global critical path. Thus the assumption that ρ corresponds to a global critical path is contradicted.

Lemma 9 implies the equality between globally consistent LPCPs and global critical paths, i.e. there is no globally consistent LPCP iff there is no global critical path. Then from theorem 5, we can obtain the major result for our distributed framework.

Theorem 10. The fault f is diagnosable in a system G iff there is no globally consistent LPCP.

4. ALGORITHM

Algorithm 1 Distributed Diagnosability Verification

1: INPUT:
   component models $G_1, ..., G_n$ of the system G;
   the considered fault f that may occur in G_f
2: $T \leftarrow ConstructALTP(G_f, f)$
3: $T \leftarrow Reduce(T)$
4: $G_S \leftarrow G_f$
5: **while** $T \neq \emptyset$ and $ConnectComp(G_S) \neq \emptyset$ **do**
6: $\quad G \leftarrow Select(ConnectComp(G_S))$
7: $\quad ADD(G_S, G)$
8: $\quad T\prime \leftarrow ConstructALTP(G)$
9: $\quad T \leftarrow Sync(T, T\prime)$
10: $\quad T \leftarrow Reduce(T)$
11: **end while**
12: **if** $T = \emptyset$ **then**
13: $\quad$ return G_S
14: **else**
15: $\quad$ return T
16: **end if**

Now we describe our algorithm to check diagnosability based on theorem 10, which is optimized in the sense that we reduce the search space as small as possible by distributing the analysis on relative ALTPs. The starting point is the construction and reduction of the ALTP of the component G_f. With the reduced ALTP containing only LPCPs, the core part is its incremental synchronization with the ALTPs of components communicating with current subsystem. Here the synchronized product of two ALTPs T_i^a and T_j^a is called the ALTP of the subsystem composed of G_i and G_j. As shown in the pseudo-code for this verification procedure, algorithm 1 performs as follows. Given the input as component models, the fault f that may occur in the component G_f, we first construct the ALTP of G_f before reducing it to retain only LPCPs (line 2, 3). Current subsystem, denoted by G_S, is then assigned by G_f. When the reduced ALTP of G_S is not empty and there exists at least one component neighboring to G_S, i.e. a component has at least one communication event in common with the ones of G_S, there exists at least one consistent LPCP of current subsystem whose global consistency should be further checked, then the algorithm repeatedly performs as follows:

(1) Select one component neighboring to G_S and then update G_S by adding this selected component. (line 6, 7)
(2) Construct the ALTP of this selected component and then synchronize it with the previous resulting reduced ALTP before reducing again this newly obtained ALTP to retain only consistent LPCPs of current subsystem G_S.(line 8-10)

Note that each time when we reduce the ALTP to retain only LPCPs, we keep the same event set. In other words, a reduced ALTP has the same event set as its corresponding non-reduced ALTP. Only in this way, we can guarantee that the synchronized result of the reduced ALTP with another ALTP based on their shared events preserves the same paths as that of their corresponding local twin plants. If the reduced ALTP of current subsystem G_S is empty, which means that there is no consistent LPCP of G_S, thus we verify the non existence of global critical paths. Then the algorithm returns current subsystem as a diagnosable subsystem. Otherwise, if there is no component connected with G_S and the reduced ALTP of G_S is not empty, then there exists at least one globally consistent LPCP. From theorem 10, the system is not diagnosable. Thus the algorithm returns the final reduced ALTP that provides some communication information about all global critical paths.

5. RESULTS DISCUSSION

When the algorithm returns a diagnosable subsystem G_S, if the number of involved components $|G_S| \leq 2$, we can directly prove that it is a minimal diagnosable subsystem. Otherwise, if $|G_S| > 2$, it is not necessarily a minimal diagnosable subsystem. When the system is diagnosable, we can enhance the possibility of the returned subsystem being a minimal diagnosable subsystem by adopting an appropriate component selection strategy. Let Σ_{S_c} be the set of communication events in current subsystem. To choose next component for further exploitation, we prefer to select the one, suppose G_i, such that $|\Sigma_{S_c} \cap \Sigma_{i_c}|$, the number of communication events in G_i contained also in the current subsystem, is maximum comparing to any other component to be selected. The idea here is to block LPCPs by the concerned communication events with as few components as possible if the system is diagnosable. In this way, more communication events of the selected component are involved in current subsystem, i.e. more communication constraints imposed on LPCPs, more likely the LPCPs may disappear after the synchronization.

Consider our example (see figure 3). Suppose that after T_1^a is built, we choose G_2 as next component to decide diagnosability, the LPCPs in T_1^a are still consistent after synchronizing with T_2^a. Thus we select G_3 for next checking and all LPCPs disappear after synchronizing with T_3^a. Then our algorithm returns a diagnosable subsystem involving all three components. However, this is not a minimal diagnosable subsystem. If we adopt component selection strategy mentioned as above, after obtaining T_1^a, we select G_3 as next component because it contains more communication events in common with the ones of G_1 (c1, c2, c3) and thus has more constraints compared to G_2, whose common communication events with G_1 is (c1, c3). When the LPCPs are synchronized with T_3^a, all of them disappear. Our algorithm thus returns a diagnosable subsystem composed of G_1, G_3, which is actually a minimal diagnosable subsystem.

Considering that the search space is exponential in the number of faults, it is better to check diagnosability by running our algorithm as many times as there are faults, each time for one

fault, which greatly reduces complexity. Obviously, our algorithm practically raises the efficiency of diagnosability problem solving. The twin plant method has polynomial space complexity in the number of system states and exponential complexity in the number of components. In our approach, from the way to construct ALTPs and to distribute diagnosability checking on them, in the worst case, the space complexity is polynomial in the number of a subset of system states, i.e. the states of communication transitions. Even though we still have exponential complexity in the number of components, but normally the growth factor is greatly reduced, i.e. the state number of ALTPs being much smaller than that of local twin plants. Furthermore, in practice, our algorithm often involves only a subset of components, both for the diagnosable cases and non-diagnosable cases.

6. RELATED WORK

In Sampath et al. (1995), the authors introduced the first definition of diagnosability for DESs and proposed a necessary and sufficient condition for testing it by constructing a deterministic diagnoser for the global system. The main drawback is its exponential space complexity in the number of system states. Jiang et al. (2001) and Yoo and Lafortune (2002) proposed new algorithms with polynomial complexity in the number of system states, which introduced the classical twin plant method. Then in Cimatti et al. (2003), symbolic model checking was proposed to test diagnosability. Thus efficiency is improved due to efficient model checking tools. However, its complexity is still exponential in the number of components.

All the above approaches have the assumption that the considered system has a monolithic model, which is not unrealistic for real complex systems. Recent work tries to solve the diagnosability problem in a distributed way to avoid global object construction (Pencolé (2004); Schumann and Pencolé (2007); Schumann and Huang (2008)). Pencolé (2004) introduces the diagnosability problem of DESs in a distributed way and solves it by synchronizing local twin plants until a global critical path is detected. In Schumann and Pencolé (2007), the proposed approach first decides non-diagnosable states in each local twin plant by propagating diagnosability information. This is done by synchronizing relative local twin plants based on their connectivity with the local twin plant of the faulty component. And then reduced local twin plants are computed that only contain the parts related to solving diagnosability problem. Then Schumann and Huang (2008) presents a scalable jointree algorithm to decide diagnosability, where diagnosability information propagation as well as consistency checking between local twin plants are both done through computing and passing messages on a jointree. Specifically, the global consistency of each local twin plant is checked by synchronizing itself with a FSM representing the behavior constraints imposed by other local twin plants. Then diagnosability can be decided on these globally consistent local twin plants.

Different from the distributed approaches mentioned above, where all their synchronization is based on local twin plants, our proposal further reduces the search space by calculating ALTPs with only necessary and sufficient diagnosability information and determines diagnosability by synchronizing the reduced ALTPs with connected ALTPs. Furthermore, we describe how to improve diagnosis algorithm with our results in a formal way when the system is verified to be diagnosable. Otherwise, our algorithm can provide the communication information about all indistinguishable behaviors resulting in non-diagnosability, which is helpful to some extent for the designer of the system to enhance its diagnosable level.

7. CONCLUSION AND FUTURE WORK

We have proposed a new theoretical framework to solve diagnosability problem considering the distributed nature of complex systems. Specifically, to check the existence of global critical paths, we begin to search for original LPCPs in T_f^a, the ALTP of the component G_f, and then check their global consistency by synchronizing with other connected ALTPs, i.e. their survival in the synchronized product. Our algorithm is at the abstract level, hiding the information at higher levels and thus resulting in a dramatic reduction in computation. Furthermore, our abstraction yields a search space as small as possible because the information in the algorithm required for diagnosability checking is minimal for a given exploration schema. But the set of components selected cannot be guaranteed as minimal when the system is diagnosable. This algorithm either returns a diagnosable subsystem or provides some information about all indistinguishable behaviors, depending on if the system is diagnosable or not. Future work is the investigation of how to minimize the observable events in a returned diagnosable subsystem by our algorithm such that it will still be diagnosable. Another perspective is to extend our distributed algorithm to deal with patterns, which can describe more general diagnosis problems, e.g. multiple faults, significant event sequences and so on (Jéron et al. (2006)).

REFERENCES

Cassandras, C.G. and Lafortune, S. (2008). *Introduction To Discrete Event Systems Second Edition*. Springer, New York.

Cimatti, A., Pecheur, C., and Cavada, R. (2003). Formal verification of diagnosability via symbolic model checking. *20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, 363–369.

Jéron, T., Marchand, H., Pinchinat, S., and Cordier, M. (2006). Supervision patterns in discrete event systems diagnosis. *Proceedings of the 8th International Workshop on Discrete Event Systems*.

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

Pencolé, Y. (2004). Diagnosability analysis of distributed discrete event systems. *Proceedings of European Conference on Articifial Intelligent ECAI'04*, 43–47.

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

Schumann, A. and Huang, J. (2008). A scalable jointree algorithm for diagnosability. *23rd American National Conference on Artificial Intelligence (AAAI-08)*.

Schumann, A. and Pencolé, Y. (2007). Scalable diagnosability checking of event-driven systems. *20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, 575–580.

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