Diagnosability Analysis for Self-observed Distributed Discrete Event Systems

Lina YE and Philippe DAGUE
Univ Paris-Sud, LRI, CNRS
Email: lina.ye@lri.fr, philippe.dague@lri.fr

***Abstract*—Diagnosability is a crucial property that determines, at design stage, how accurate any diagnosis algorithm can be on a partially observable system and, thus, has a significant impact on the performance and reliability of complex systems. Most existing approaches assumed that observable events in the system are globally observed. But, sometimes, it is not possible to obtain global information. Thus, a recent work has proposed a new framework to check diagnosability in a system where each component can only observe its own observable events to keep the internal structure private in terms of observations. However, the authors implicitly assume that local paths in components can be exhaustively enumerated, which is not true in a general case where there are embedded cycles. In this paper, we get some new results about diagnosability in such a system, i.e., what we call joint diagnosability in a self-observed distributed system. First, we prove the undecidability of joint diagnosability with unobservable communication events by reducing Post's Correspondence Problem to an observation problem. Then, we propose an algorithm to check a sufficient but not necessary condition of joint diagnosability. Finally, we briefly discuss about the decidable case with observable communication events.**

***Index Terms*—diagnosis; joint diagnosability; finite state machine.**

I. INTRODUCTION

Over the latest decades, with more performance requirements imposed on the complex systems, they are subject to more errors. However, it is unrealistic to detect faults manually for such systems. Automated diagnosis mechanisms are thus required for large distributed applications. Generally speaking, diagnosis reasoning aims at detecting possible faults explaining the observations. The efficiency of diagnosis reasoning depends on system diagnosability, which is a crucial property that determines at design stage how accurate any diagnosis algorithm can be on a partially observable system. The systems we discuss here are Discrete Event Systems (DES).

Some existing works analyzed diagnosability in a centralized way ([1], [2], etc.), i.e., a monolithic model of a given system is hypothesized, which is unrealistic due to combinatorial explosion of state space. This is why very recently distributed approaches began to be investigated ([3], [4], etc.), relying on local objects. However, all these approaches assumed that observable events in the system are globally observed. But sometimes it is not possible to obtain global information. Then, Ye et al. [5] has proposed a new framework to check diagnosability in a system where each component can only observe its own observable events to keep the internal structure private in terms of observations. However, the authors implicitly assume that local paths can be exhaustively enumerated, which is not true in a general case where there are embedded cycles. In this paper, we generalize this work to get some new results about the diagnosability of what we call self-observed distributed systems, where observable events can only be observed by their own component.

We make several contributions in this paper. First, we extend diagnosability of globally observed systems to what we call joint diagnosability of self-observed systems and then to prove its undecidability with unobservable communication events. Secondly, we propose an algorithm for testing a sufficient condition, where we obtain pairs of local trajectories in the faulty component, such that for each pair only one trajectory contains the fault but both have the same observations, and then check their global consistency through two phases. We provide the proof that it is a sufficient condition and point out why it is not necessary. Thirdly, the decidable case where communication events are observable is discussed.

II. PRELIMINARIES

In this section, we model self-observed distributed DES and then recall joint diagnosability features [5].

We consider a self-observed distributed DES composed of a set of components $\{G_1, G_2, ..., G_n\}$ that communicate by communication events, where each component can only observe its own observable events. Such a system is modeled by a set of finite state machines (FSM), each one representing the local model of one component. The local model of a component G_i is 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; and q_i^0 is the initial state. The set of events Σ_i is partitioned into four subsets: Σ_{i_o}, the set of locally observable events that can be observed only by their own component G_i; Σ_{i_u}, the set of unobservable normal events; Σ_{i_f}, the set of unobservable fault events; and Σ_{i_c}, the set of communication events shared by at least one other component, which are the only shared events between components. Figure 1 depicts a self-observed distributed system with two components: G_1 (left) and G_2 (right), where the events Oi denote locally observable events, the event F denotes an unobservable fault event, the events Ui denote unobservable normal events and the events Ci denote communication events.

We denote the synchronized FSM of components $G_1, ..., G_n$ by $\|(G_1, ..., G_n)$, where the synchronized events are the shared events between components and any one of them

 ISBN: 978-1-61208-233-2

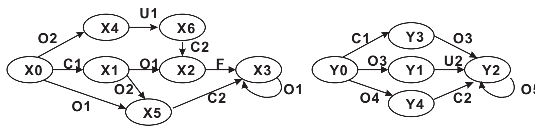

Fig. 1. A system with two components G_1 (left) and G_2 (right).

always occurs simultaneously in all components that define it. The global model of the entire system is implicitly defined as the synchronized FSM of all components based on their shared events, i.e., communication events. However, the global model will not be calculated since in a self-observed distributed system, the global occurrence order of observable events is not accessible. In the following, we call subsystem of G the synchronization of a subset of components of G, i.e., $\|(G_{s_1}, ..., G_{s_m})$, where $\{s_1, ...s_m\} \subseteq \{1, ...n\}$. One component or the entire system can be considered as a subsystem.

Given a system model $G = (Q, \Sigma, \delta, q^0)$, the set of words produced by the FSM G is a prefix-closed language $L(G)$ that describes the normal and faulty behaviors of the system. Formally, $L(G) = \{s \in \Sigma^* | \exists q \in Q, (q^0, s, q) \in \delta\}$, where the transition δ has been extended from events to words. In the following, we call a word of $L(G)$ also a **trajectory** in the system G and a sequence $q_0\sigma_0q_1\sigma_1...$ a **path** in G, where $\sigma_0\sigma_1...$ is a trajectory and for all i, we have $(q_i, \sigma_i, q_{i+1}) \in \delta$. Given $s \in L(G)$, we denote the post-language of $L(G)$ after s by $L(G)/s$ and denote the projection of s to observable events of G (resp. G_i) by $P(s)$ (resp. $P_i(s)$). We adopt the assumption in [3], i.e., the projection of the global language on each local model is observable live, i.e., there is no unobservable cycle in any component. For the sake of simplicity, our approach is shown for only one fault, which can be extended to the case with multiple faults by running one time for each fault. Next we rephrase reconstructibility introduced in [7].

Definition 1: (Reconstructibility). Given a system G that is composed of several subsystems, i.e., $G = \|(G_{s_1}, ..., G_{s_m})$, a set of trajectories in these subsystems is said to be reconstructible with respect to G if it is obtained by projection on this set of subsystems of a trajectory in G.

If there is no common communication event between two subsystems, then any trajectory in one subsystem and any one in the other subsystem are reconstructible.

For the sake of consistency, now we rename what is called cooperative diagnosability in [5] as joint diagnosability. We denote a trajectory ending with the fault f by s^f.

Definition 2: (Joint diagnosability). A fault f is jointly diagnosable in a self-observed distributed system G with components $\{G_1, ...G_n\}$, iff

$$\exists k \in N, \forall s^f \in L(G), \forall t \in L(G)/s^f, (\forall i \in \{1, ..., n\}, |P_i(t)| \geq k) \Rightarrow (\forall p \in L(G)\ (\forall i \in \{1, ..., n\}, P_i(p) = P_i(s^f.t)) \Rightarrow f \in p).$$

Joint diagnosability of f means that for each faulty trajectory s^f in G, for each extension t with enough locally observable events in all components, every trajectory p in G that is equivalent to $s^f.t$ for local observations in each component should contain in it f. In other words, the fault can be detected after finite non bounded trajectory prolongation in at least one component. In a self-observed system, we call a pair of trajectories p and $p\prime$ satisfying the three conditions a (global) **indeterminate pair**: 1) p contains f and $p\prime$ does not; 2) p has arbitrarily long local observations in all components after the occurrence of f; 3) $\forall i \in \{1, ..., n\}$, $P_i(p) = P_i(p\prime)$. Here arbitrarily long local observations can be considered as infinite local observations. Now we have the following theorem [5].

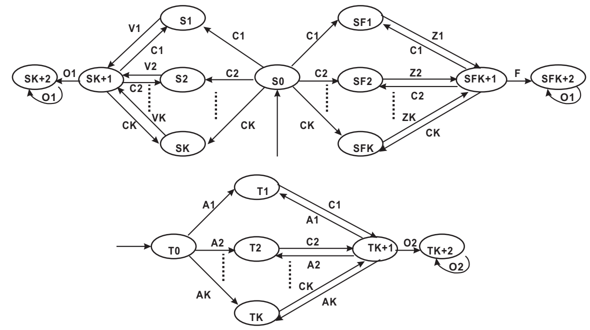

Fig. 2. A system with two components G_1 (top) and G_2 (bottom) for proving undecidability of joint diagnosability.

Theorem 1: Given a self-observed distributed system G, a fault f is jointly diagnosable in G iff there is no (global) indeterminate pair in G.

III. UNDECIDABLE CASE

To discuss about joint diagnosability, we consider two cases: communication events being unobservable and observable. We first consider the general case, i.e., communication events being unobservable.

Theorem 1 implies that checking joint diagnosability boils down to check the existence of indeterminate pairs that witnesses non joint diagnosability. Inspired from [6], where undecidability of joint observability is proved by reducing the Post's Correspondence Problem (PCP) to an observation problem, we discuss first about whether joint diagnosability is decidable or not.

For the sake of simplicity, we give now a simplified proof for undecidability of joint diagnosability.

Theorem 2: Given a self-observed distributed system where communication events are unobservable, checking joint diagnosability of a given fault is undecidable.

Proof:

1) PCP: given a finite alphabet Σ, two sets of words $v_1, v_2, ..., v_k$ and $z_1, z_2, ..., z_k$ over Σ, then a solution to PCP is a sequence of indices $(i_m)_{1 \leq m \leq n}$ with $n \geq 1$ and $1 \leq i_m \leq k$ for all m such that $v_{i_1}v_{i_2}...v_{i_n} = z_{i_1}z_{i_2}...z_{i_n}$.

2) Now consider the example depicted in Figure 2, where the system is composed of two components G_1 and G_2. In G_1, each one of $Vi, i \in \{1, ..., k\}$, and each one of $Zi, i \in \{1, ..., k\}$, denotes a sequence of observable events all different from $O1$, $C1, ..., Ck$ are unobservable communication events, F denotes a fault event and $O1$ is an observable event. In G_2, each one of $Ai, i \in \{1, ..., k\}$, denotes an observable event different from $O2$, $C1, ..., Ck$ are unobservable communication events and $O2$ is an observable event. Then the observations in G_1 can be described as $Vi_1Vi_2...Vi_nO1^*$ without fault or

$Zi_1Zi_2...Zi_nO1^*$ with fault, where $\forall i_j, j \in \{1,...,n\}, i_j \in \{1,...,k\}$. In G_2, the observations are $Ai_1Ai_2...Ai_nO2^*$. In this system, the occurrence of the fault can be confirmed by the observation of $O1$.

3) Without the observation of $O1$, the local observations are $wO1^+$ for G_1 and $Ai_1Ai_2...Ai_nO2^*$ for G_2, where $w = Vi_1Vi_2...Vi_n$ when there is no fault or $w = Zi_1Zi_2...Zi_n$ when there is a fault. Clearly, if PCP has a solution, i.e., $\exists(i_m)_{1\leq m\leq n}$ such that $Vi_1Vi_2...Vi_n = Zi_1Zi_2...Zi_n$, we have two trajectories p and $p\prime$ such that the observations of p in G_1 are $Vi_1Vi_2...Vi_nO1^+$, which is a trajectory without fault, while the observations of $p\prime$ in G_1 are $Zi_1Zi_2...Zi_nO1^+$, which is a trajectory with a fault. And both p and $p\prime$ have the same observations for G_2, i.e., $Ai_1Ai_2...Ai_nO2^*$. Thus we get that p and $p\prime$ have the same observations for both G_1 and G_2, i.e., $Vi_1Vi_2...Vi_nO1^+$=$Zi_1Zi_2...Zi_nO1^+$ for G_1 and $Ai_1Ai_2...Ai_nO2^*$ for G_2, then the fault is not jointly diagnosable.

4) On the other hand, if the fault is not jointly diagnosable, then we obtain at least one indeterminate pair, denoted by p and $p\prime$ such that the projection of p on G_1 is $Ci_1Vi_1Ci_2Vi_2...Ci_nVi_nO1^*$, on G_2 is $Ai_1Ci_1Ai_2Ci_2...Ai_nCi_nO2^*$ and that of $p\prime$ on G_1 is $Cj_1Zj_1Cj_2Zj_2...Cj_mZj_mFO1^*$ and on G_2 is $Aj_1Cj_1Aj_2Cj_2...Aj_mCj_mO2^*$. From the fact that p and $p\prime$ have the same observations for G_2, we get $Ai_1Ai_2...Ai_nO2^* = Aj_1Aj_2...Aj_mO2^*$ and thus we have $m = n$ and $i_1 = j_1, ..., i_n = j_n$. And then from the same observations of p and $p\prime$ on G_1, we get $Vi_1Vi_2...Vi_nO1^* = Zi_1Zi_2...Zi_nO1^*$, i.e., $Vi_1Vi_2...Vi_n = Zi_1Zi_2...Zi_n$, which means that there is a solution for PCP.

5) The above proves that the existence of a solution for PCP is equivalent to that of a fault being not jointly diagnosable. Since PCP is an undecidable problem, then checking joint diagnosability is undecidable. ■

There are two major differences between joint diagnosability in our framework and joint observability in [6]. One is that the former assumes that local observers are attached to local components that are synchronized by common communication events to get a global model while the latter separates arbitrarily the observable events in the global model into several sets. The other one is that joint diagnosability consists in separating infinite trajectories while joint observability consists in separating finite ones. Thus, if any communication event is assumed to be unobservable, joint diagnosability checking boils down to infinite PCP. But this one has also been proved to be undecidable [8], which gives the result.

IV. SUFFICIENT ALGORITHM

We have proved that joint diagnosability with unobservable communication events is undecidable. We can nevertheless propose an algorithm to test a sufficient condition, which is still quite useful in some circumstances. We first construct the local diagnoser from a given local model to show fault information for any local trajectory. Then, we show how to build a structure called local twin plant to obtain original information about indeterminate pairs (also called local indeterminate pairs in the following), based on the local diagnoser. The next step is to check the global consistency, i.e., to check whether the local indeterminate pairs can be extended into (global) indeterminate pairs, whose existence testifies non joint diagnosability. Actually, our algorithm remains trivially applicable when the assumption of unobservability of communication events is partially relaxed, i.e., in the most general case where some communication events are observable and others unobservable.

A. Original diagnosability information

To check the existence of indeterminate pairs, in the distributed framework, we use the structure called local twin plant defined in [2]. In particular, the considered fault is assumed to only occur in one component, denoted by G_f. Then the local twin plant for G_f contains original information of indeterminate pairs: actually this twin plant is a FSM that compares every pair of local trajectories to search for the pairs with the same arbitrarily long local observations, but exactly one of the two containing a fault, which are local indeterminate pairs. First, we define delay closure operation with respect to a subset Σ_d of Σ to preserve only the information about the events in Σ_d.

Definition 3: (Delay Closure). Given a FSM G = (Q, Σ, δ, q^0), its delay closure with respect to $\Sigma_d \subseteq \Sigma$ is $\complement_{\Sigma_d}(G) = (Q, \Sigma_d, \delta_d, q^0)$ where $(q, \sigma, q') \in \delta_d$ iff $\exists s \in (\Sigma\backslash\Sigma_d)^*, (q, s\sigma, q') \in \delta$.

We now describe how to construct the local diagnoser of a given component, based on which we build the local twin plant. Given a local model, we get a modified one by attaching fault label, denoted by $l \in \{N, F\}$, where N for normal and F for fault, to each state. In other words, before the occurrence of the fault, each state is labeled with label N and, after its occurrence, with label F.

Definition 4: (Local diagnoser). Given a local model G_i, its local diagnoser D_i is obtained by operating the delay closure with respect to the set of communication events and observable events on the modified model: $D_i = \complement_{\Sigma_{i_o}\cup\Sigma_{i_c}}(G_i^m)$, where G_i^m is the modified version of G_i.

Based on the local diagnoser, the corresponding local twin plant is obtained by synchronizing the local diagnoser with itself based on the locally observable events, allowing one to obtain all pairs of local trajectories with the same observations to search for local indeterminate pairs. To simplify this synchronization, the two identical local diagnosers, denoted by D_i^l (left instance) and D_i^r (right instance), can be reduced as follows: D_i^l is obtained by retaining only paths with at least one fault cycle and D_i^r is obtained by retaining only paths with at least one non-fault cycle. This reduction keeps all original diagnosability information since what we are interested in are only local indeterminate pairs. However, this reduction is only applicable for the local diagnoser of the faulty component G_f; for other components, the local twin plant is obtained by synchronizing the non reduced left instance and the non reduced right instance since there is no fault information. Since this synchronization is based on observable events Σ_{i_o}, the

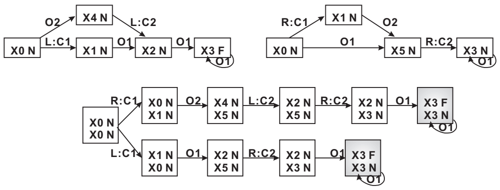

Fig. 3. Two reduced instances of the diagnoser for G_1 (top) and part of the corresponding local twin plant (bottom).

non-synchronized events are distinguished by the prefix L or R: in D_i^l (D_i^r), each communication event $c \in \Sigma_{i_c}$ from D_i is renamed by $L : c$ ($R : c$). The names of all locally observable events are left unchanged.

Definition 5: (Local twin plant). Given a local diagnoser D_i for the component G_i, the corresponding local twin plant is a FSM, denoted by $T_i = D_i^l \| D_i^r$, where the synchronized events are locally observable events in G_i.

Each state of a local twin plant is a pair of local diagnoser states providing two possible diagnoses with the same local observations. Given a twin plant state $((q^l, l^l)(q^r, l^r))$, if the considered fault $f \in l^l \cup l^r$ but $f \notin l^l \cap l^r$, which means that the occurrence of f is not certain up to this state, then this state is called an ambiguous state with respect to the fault f. An ambiguous state cycle is a cycle containing only ambiguous states. In a local twin plant, if a path contains an ambiguous state cycle with at least one locally observable event, then it is called a **local indeterminate path**, which corresponds to a local indeterminate pair. Note that local indeterminate paths contain original diagnosability information and can be obtained only in the local twin plant of the component G_f. If a local indeterminate pair can be extended into a global indeterminate pair, then we say that its corresponding local indeterminate path is globally consistent. Figure 3 shows the left and right instances of the local diagnoser for the faulty component G_1 of Figure 1 (top) as well as a part of the corresponding local twin plant (bottom). Clearly, in the local twin plant, we have local indeterminate paths since they have ambiguous state cycles with observable events.

B. Global consistency checking

Joint diagnosability verification consists in checking the existence of globally consistent local indeterminate paths, whose existence proves non joint diagnosability. To do this, we have to check the global consistency of the corresponding left trajectories of the local indeterminate paths in the local twin plants as well as that of their corresponding right trajectories, shortly called left consistency checking and right consistency checking.

Definition 6: (Left (Right) consistent plant). Given a subsystem G_S composed of components $G_{i_1}, ..., G_{i_m}$ and their corresponding local twin plants $T_{i_1}, ..., T_{i_m}$, to obtain a left (right) consistent plant with respect to the subsystem G_S, denoted by T_f^l (T_f^r), we perform the following two steps:
1) Distinguish right (left) communication events between local twin plants by renaming them with the prefix of component ID. For example, $R{:}C2$ ($L{:}C2$) in the local twin plant of G_2 is renamed as $G_2{:}R{:}C2$ ($G_2{:}L{:}C2$).

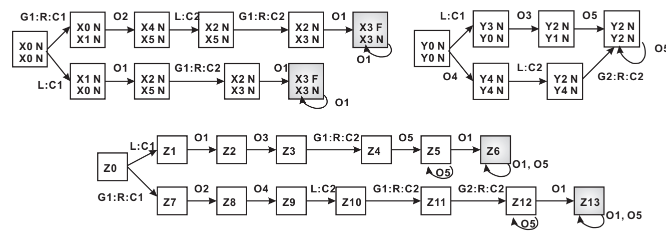

Fig. 4. Part of the renamed local twin plants for G_1 and G_2 (top) and part of the left consistent plant T_f^l (bottom).

2) Synchronize the renamed local twin plants with the synchronized events being the common left (right) communication events, which works because observable events do not intersect between components and non-synchronized right (left) communication events are distinguished by the prefix of component ID.

From definition 1, we know that in the left (right) consistent plant with respect to a subsystem G_S, each path p corresponds to a set of paths $p_{i_1}, ..., p_{i_m}$ in the local twin plants of all components in G_S such that the set of left (right) trajectories of $p_{i_1}, ..., p_{i_m}$ are reconstructible with respect to G_S. For our example, the bottom part of Figure 4 shows a part of the left consistent plant T_f^l, which is obtained by synchronizing the renamed local twin plant of G_1 and that of G_2 (top part of Figure 4) based on the common left communication events.

C. Algorithm

Algorithm 1 presents the procedure to verify a sufficient condition of joint diagnosability. As shown in the pseudo-code, algorithm 1 performs as follows. Given the input as the set of component models, the fault F that may occur in the component G_f, we initialize the parameters as empty, i.e., G_S^l (G_S^r), the subsystem for the left (right) consistency checking. The procedure of the algorithm can be separated into two parts: left consistency checking (line 3-12) and right consistency checking (line 13-24).

Left consistency checking begins with the local twin plant construction of G_f, the subsystem G_S^l being now G_f (line 3-4). When both the left consistent plant T_f^l with respect to the current left subsystem G_S^l and $DirectCC(G, G_S^l)$ are not empty (line 5), where $DirectCC(G, G_S^l)$ is the set of directly connected components to G_S^l (a directly connected component being one sharing at least one common communication event with the subsystem), the algorithm repeatedly performs the following steps to further check left consistency.
1) Select one directly connected component G_i to the subsystem G_S^l and construct its local twin plant T_i (line 6-7).
2) Synchronize T_f^l with T_i to obtain left consistent plant for this extended subsystem based on common left communication events (line 8). To do this, non-synchronized right communication events are distinguished by the prefix of component ID.
3) Update the subsystem G_S^l by adding G_i and reduce the newly obtained T_f^l by retaining only paths with ambiguous state cycles containing observable events for all components in G_S^l (line 9-10).

Algorithm 1 Sufficient algorithm

1: INPUT: the system model $G = (G_1, ..., G_n)$; the fault F and the faulty component G_f
2: Initializations: $G_S^l \leftarrow \emptyset$ (subsystem for left consistency checking); $G_S^r \leftarrow \emptyset$ (subsystem for right consistency checking)
3: $T_f^l \leftarrow ConstructLTP(G_f)$
4: $G_S^l \leftarrow G_f$
5: **while** $T_f^l \neq \emptyset$ and $DirectCC(G, G_S^l) \neq \emptyset$ **do**
6: $\quad G_i \leftarrow SelectDirectCC(G, G_S^l)$
7: $\quad T_i \leftarrow ConstructLTP(G_i)$
8: $\quad T_f^l \leftarrow T_f^l \| T_i$
9: $\quad G_S^l \leftarrow Add(G_S^l, G_i)$
10: $\quad T_f^l \leftarrow RetainConsisPaths(T_f^l)$
11: **if** $T_f^l = \emptyset$ **then**
12: $\quad$ return "F is jointly diagnosable in G"
13: **else**
14: $\quad T_f^r \leftarrow AbstractRight(G_f, T_f^l)$
15: $\quad G_S^r \leftarrow G_f$
16: $\quad$ **while** $T_f^r \neq \emptyset$ and $G_S^l \neq G_S^r$ **do**
17: $\quad\quad G_i \leftarrow SelectDirectCC(G_S^l, G_S^r)$
18: $\quad\quad T_f^r \leftarrow T_f^r \| AbstractRight(G_i, T_f^l)$
19: $\quad\quad G_S^r \leftarrow Add(G_S^r, G_i)$
20: $\quad\quad T_f^r \leftarrow RetainConsisPaths(T_f^r)$
21: $\quad$ **if** $T_f^r = \emptyset$ **then**
22: $\quad\quad$ return "F is jointly diagnosable in G"
23: $\quad$ **else**
24: $\quad\quad$ return "Joint diagnosability cannot be determined"

If the left consistent plant T_f^l is empty, then there is no local indeterminate path that corresponds to a set of paths in the local twin plants of all components in the subsystem such that their left trajectories are reconstructible (definition 1), which implies the non existence of a globally consistent local indeterminate path. In this case joint diagnosability information is returned (line 11-12). Otherwise, if T_f^l is not empty (line 13), then we proceed to check right consistency of the corresponding paths in T_f^l that have been already verified to be left consistent in the whole system.

Right consistency checking begins with the function $AbstractRight(G_f,\ T_f^l)$ (line 14), which performs delay closure with respect to right communication events and observable events of G_f. Then the subsystem G_S^r is assigned as G_f (line 15). When the right consistent plant T_f^r for the current right subsystem G_S^r is not empty and $G_S^l \neq G_S^r$ (line 16), we repeatedly perform the following steps to check right consistency in an extended subsystem (since left consistency checking does explore all connected components, for right consistency checking we only consider the subsystem G_S^l instead of the whole system).
1) Select a directly connected component G_i to G_S^r from G_S^l (line 17).
2) Perform the function $AbstractRight(G_i,\ T_f^l)$, which has been described as above, and then synchronize with T_f^r based on the common right communication events (line 18). To do this, we rename the right communication events by removing the prefix of component ID, e.g., $G_i{:}R{:}C2$ renamed as $R{:}C2$.
3) Update the subsystem G_S^r by adding G_i and reduce the newly obtained T_f^r by retaining only paths with ambiguous state cycles containing observable events for all components in G_S^r (line 19-20).

If the right consistent plant T_f^r is empty, then there is no local indeterminate path that corresponds to a set of paths in the local twin plants such that their left trajectories and right trajectories are reconstructible respectively, i.e., there is no globally consistent local indeterminate path. In this case, the algorithm returns joint diagnosability information (line 21-22). Otherwise, if T_f^r is not empty, we cannot determine whether the fault is jointly diagnosable or not. Then the algorithm returns indetermination information (line 23-24). In other words, empty left consistent plant T_f^l or empty right consistent plant T_f^r is a sufficient condition but not a necessary condition of joint diagnosability.

Theorem 3: In algorithm 1, if the left consistent plant T_f^l or the right consistent plant T_f^r is empty, then the fault is jointly diagnosable, but the reverse is not true.

Proof: :
($\Rightarrow$) Suppose that T_f^l or T_f^r is empty and that the fault is not jointly diagnosable. From non joint diagnosability, it follows that there exists at least one globally consistent local indeterminate path. Since global consistency of a local indeterminate path implies both left consistency and right consistency, from algorithm 1 we know that, after left and right consistency checking, this local indeterminate path must correspond to a path both in T_f^l and in T_f^r. Thus neither T_f^l nor T_f^r is empty, which contradicts the assumption.
($\not\Leftarrow$) Now we explain why non emptiness of both T_f^l and T_f^r does not necessarily imply that the fault is not jointly diagnosable. Suppose that T_f^l is not empty and that it contains two paths, denoted by $\rho 1$ and $\rho 2$, corresponding to two local indeterminate paths. $\rho 1$ corresponds to a set of paths $\rho_i^1, 1 \leq i \leq n$ in the local twin plants of all components and $\rho 2$ corresponds to a set of paths $\rho_i^2, 1 \leq i \leq n$ in all local twin plants. Now suppose that the right trajectories of the set of paths $\rho_i^1, 1 \leq i \leq n$ are not reconstructible and the same for that of the set of paths $\rho_i^2, 1 \leq i \leq n$. It follows that the two local indeterminate paths cannot be extended into global indeterminate pairs and thus are not globally consistent. Then we further suppose that the right trajectories of the set of paths $\rho_1^1, ..., \rho_{n-1}^1, \rho_n^2$ are reconstructible or the same for the set of paths $\rho_1^2, ..., \rho_{n-1}^2, \rho_n^1$. In this case, from algorithm 1, it follows that finally the right consistent plant T_f^r is not empty. Now both T_f^l and T_f^r are not empty but there is no globally consistent local indeterminate paths, i.e., the fault is jointly diagnosable. ■

Now, we illustrate on our example the fact that the condition is not necessary. The top part of Figure 5 shows the results of performing delay closure with respect to right communication events and observable events both for

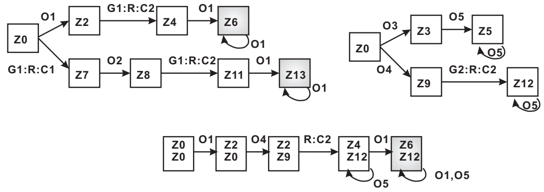

Fig. 5. FSM after delay closure on the left consistent plant (Figure 4) for G_1 and G_2 (top) and part of the right consistent plant (bottom).

G_1 and G_2 on the left consistent plant depicted in Figure 4. Then, to check right consistency, we rename again the right communication events by removing the component ID such that they can be synchronized. The bottom part of Figure 5 shows a part of the right consistent plant, which is not empty. Now, both left and right consistent plants are not empty, but this does not imply the existence of global indeterminate pairs that witnesses non joint diagnosability. Actually, the part of the left consistent plant depicted here corresponds to two local indeterminate pairs in G_1 with their corresponding left consistent pairs in G_2, i.e., one local indeterminate pair is $((C1.O1.F.O1^*), (O1.C2.O1^*))$ in G_1 with its left consistent pair $((C1.O3.O5^*), (O3.U2.O5^*))$ in G_2 and the other local indeterminate pair is $((O2.U1.C2.F.O1^*), (C1.O2.C2.O1^*))$ in G_1 with its left consistent pair $((O4.C2.O5^*), (O4.C2.O5^*))$ in G_2. While the right consistent plant shown here corresponds to one local indeterminate pair in G_1, which is $((C1.O1.F.O1^*), (O1.C2.O1^*))$, with its right consistent pair in G_2, i.e., $((O4.C2.O5^*), (O4.C2.O5^*))$. Thus, we can see that the same local indeterminate pair does not correspond to the same consistent pair in G_2 in the left consistent plant and in the right consistent plant, which means that this local indeterminate pair cannot be extended into a global indeterminate pair. Our algorithm gives indeterminate information for joint diagnosable systems that satisfy the following condition: for any set of paths including one path in the local twin plant of each non faulty component and one local indeterminate path in that for faulty component, if they are left consistent and right consistent respectively, then their corresponding local trajectories in the components cannot constitute an indeterminate pair through synchronization. Our illustrated example is quite tricky to show the possibility of indeterminate decision given by our algorithm for a joint diagnosable system. However, in reality, a system satisfying the above condition is quite rare and thus this algorithm can be applicable for a large number of complex systems.

V. DECIDABLE CASE

We have proved the undecidability of joint diagnosability with unobservable communication events. If we assume their observability, then this problem becomes decidable. Because when any communication event is observable, in the local twin plant, we obtain all pairs of local trajectories with the same observations, including the same observable communication events. Thus, each path in the local twin plant corresponds to a pair of local trajectories with the same sequence of communication events. It follows that, during global consistency checking, the separate checkings for left and right consistency becomes now only one checking. While in algorithm 1, the checking into two separate phases is the reason why it gives only a sufficient but not necessary condition. Actually, the observability of communication events makes joint diagnosability equivalent to classical diagnosability since only one checking for global consistency implies the same global occurrence order of observations for global indeterminate pairs.

VI. CONCLUSION AND FUTURE WORK

In this paper, we consider self-observed distributed systems where observable events can only be observed by their own component and thus the distributed and private (w.r.t. observation) nature of real systems is taken into account. Then, we prove the undecidability of joint diagnosability checking when communication events are unobservable, before proposing an algorithm to test a sufficient condition. We start from local indeterminate paths and then we check both in sequence left consistency and right consistency. Due to the observation-privacy, the global occurrence order of observable events between different components is not known, which is taken into account through constructing left and right consistent plants separately. For computational complexity, as distributed diagnosability approaches with globally observable events, in the worst case, our algorithm has polynomial complexity in the number of system states and exponential complexity in the number of system components. But our approach is more autonomous thanks to distributed observations. Then we briefly discuss the decidable case where communication events are observable. There is a gap between these two cases as the unobservable case is undecidable and the observable case is decidable. Next interesting work is to investigate where is the frontier between the two cases, i.e., to study the decidability of joint diagnosability for partial observability of communication events.

REFERENCES

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

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

[3] Y. Pencolé, "Diagnosability analysis of distributed discrete event systems," in *Proceedings of 16th European Conference on Artificial Intelligence ECAI'04*, Valencia, Spain, August 2004, pp. 43–47.

[4] A. Schumann and Y. Pencolé, "Scalable diagnosability checking of event-driven systems," in *Proceedings of 20th International Joint Conference on Artificial Intelligence IJCAI-07*, Hyderabad, India, 2007, pp. 575–580.

[5] L. Ye and P. Dague, "Diagnosability analysis of discrete event systems with autonomous components," in *Proceedings of 19th European Conference on Artificial Intelligence ECAI-10*, Lisbon, Portugal, August 2010, pp. 105–110.

[6] S. Tripakis, "Undecidable problems of decentralized observation and control," in *40th IEEE Conference on Decision and Control*, Orlando, Florida, December.

[7] R. Cori and Y. Métivier, "Recognizable subsets of some partially abelian monoids," *Theoretical Computer Science*, vol. 35, pp. 179–189, 1985.

[8] V. Halava and T. Harju, "Undecidability of infinite post correspondence problem for instances of size 9," *Theoretical Informatics and Applications*, vol. 40, pp. 551–557, 2006.