Scalable Diagnosability Checking of Event-Driven Systems*

Anika Schumann
The Australian National University,
National ICT Australia
anika.schumann@anu.edu.au

Yannick Pencolé
National Center for Scientific Research
LAAS-CNRS, France
yannick.pencole@laas.fr

Abstract

Diagnosability of systems is an essential property that determines how accurate any diagnostic reasoning can be on a system given any sequence of observations. Generally, in the literature of dynamic event-driven systems, diagnosability analysis is performed by algorithms that consider a system as a whole and their response is either a positive answer or a counter example. In this paper, we present an original framework for diagnosability checking. The diagnosability problem is solved in a distributed way in order to take into account the distributed nature of realistic problems. As opposed to all other approaches, our algorithm also provides an exhaustive and synthetic view of the reasons why the system is not diagnosable. Finally, the presented algorithm is scalable in practice: it provides an approximate and useful solution if the computational resources are not sufficient.

1 Introduction

Nowadays, monitoring and diagnosing complex communicating event-driven systems has huge economic impact by ensuring better reliability, maintenance and safety [Lamperti and Zanella, 2003]. Such diagnostic capabilities are, for instance, required in aeronautic [Ghelam *et al.*, 2006] or automotive systems, telecommunications [Pencolé and Cordier, 2005] and electronic commerce [Yan *et al.*, 2005]. For many years, the problem of automated fault diagnosis of these kinds of systems has received constant and considerable attention from researchers in the fields of Artificial Intelligence and Control. Given a monitor continuously receiving observations from a system, automated diagnosis methods aim at detecting all possible faults that explain the observations.

However, the resulting diagnosis depends on the diagnosability of the system [Sampath *et al.*, 1995]. If the system is strongly diagnosable, a diagnostic process will find an accurate explanation for any possible set of observations from the system, otherwise the diagnosis will provide an ambiguous and useless explanation. Consequently, diagnosability analyses should be performed on the system before any diagnostic reasoning. The diagnosability results then help in choosing the type of diagnostic algorithm that can be performed and provide some information of how to change the system to make it more diagnosable [Pencolé, 2004].

In this paper, we propose a formal framework for checking diagnosability on event-driven systems which is mainly motivated by two facts. On the one hand, checking diagnosability means determining the existence of two behaviours in the system that are not *distinguishable*. However, in realistic systems, there is a combinatorial explosion of the search space that forbids the practical use of classical and centralised diagnosability checking methods [Sampath *et al.*, 1995] like the twin plant method [Jiang *et al.*, 2001; Yoo and Lafortune, 2002]. On the other hand, in the case of a nondiagnosable system, checking its diagnosability is not sufficient; the diagnosability analysis should also provide the reasons why the system is not diagnosable. Given these reasons, either the system specification is redesigned or diagnostic reasoning is improved by taking into account the fact that the diagnosis of a fault may always be ambiguous.

Our proposal makes three contributions to the diagnosability problem. The first one is the definition of a new theoretical framework where the diagnosability problem is described as a distributed search problem. Instead of searching for not distinguishable behaviours in a global twin plant, we propose to distribute the search based on local twin plants and to determine such global behaviours without computing any part of the global twin plant. The second contribution comprises the nature of the feedback in case the faults are not diagnosable. Since the search is distributed over the whole system, the analysis provides an exhaustive and synthetic view of the diagnosability analysis of the system (why a given fault is not diagnosable). The third contribution is about the practical use of the algorithm. Since the diagnosability analysis problem is complex, the complete analysis may not be possible due to a lack of computational resources. However, the distributed search makes the computation scalable in the sense that our algorithm is able to provide an approximate but exhaustive solution to the diagnosability problem whatever the computational resources are.

The paper is organised as follows. Section 2 presents some

*This research was supported by National ICT Australia (NICTA) in the framework of the SuperCom project (Model-Based Supervision of Composite Systems). NICTA is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian National Research Council.

background on diagnosability on event-driven systems. Section 3 introduces our theoretical framework for diagnosability checking. Section 4 presents a scalable algorithm for distributed diagnosability checking. In section 5, related works are compared and discussed.

2 Background

2.1 Diagnosability in discrete-event systems

The system is modelled using classical automata: this formalism is aimed at modelling any discrete event system with multiple faults [Sampath *et al.*, 1995]. We call a *fault* or a *critical event* any event whose occurrence a monitoring agent aims to identify. We suppose that a fault is not observable and occurs in one component. The modelling of the system is modular: each component has a *local model* and a synchronisation operator $Sync$ is defined in order to implicitly represent the global model of the system.

Definition 1 (Model of a component) *The model of the component i is the finite state machine $G_i = \langle X_i, \Sigma_i, x_{0_i}, T_i \rangle$, where*

- *X_i is the set of states ($X_i = \{x_{1_i}, \ldots, x_{m_i}\}$),*
- *Σ_i is the set of events,*
- *x_{0_i} is the initial state, and*
- *T_i is the transition set ($T_i \subseteq X_i \times \Sigma_i \times X_i$).*

The set of events Σ_i is divided into four disjoint parts: Σ_{o_i} is the set of observable events, Σ_{s_i} is the set of events shared among components (they represent communications between components), Σ_{f_i} is the set of unobservable fault events and Σ_{u_i} is the set of other unobservable events.

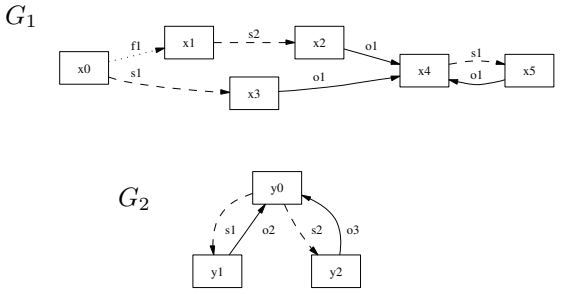

Figure 1: Model of a system defined by two components. Solid lines denote observable transitions, dashed lines shared transitions and dotted lines failure transitions.

Figure 1 illustrates a system composed of two local models defined as above. The synchronisation operation, denoted $Sync(\{M_1, \ldots, M_n\}, \mathcal{S})$, is the classical synchronisation operation on the n finite state machines $\{M_i\}$ based on the set of shared events $\mathcal{S}$. The result M of the synchronisation is the subset of the Cartesian product $M_1 \times \ldots \times M_n$ restricted with the following rules.

- If $e \in \mathcal{S}$ then from any state $(x_1, \ldots, x_n)$, the event e can occur if for all machines M_j where e can occur, there exists in M_j a transition $x_j \xrightarrow{e} x'_j$.
- If $e \notin \mathcal{S}$ then from any state $(x_1, \ldots, x_n)$, the event e can occur if there exists a machine M_j where there exists a transition $x_j \xrightarrow{e} x'_j$.

The models of components and the operation $Sync$ define the decentralized model of the system. The *global model* G then results from the synchronised product of the n components $\{G_i\}$ where the shared events are synchronised:

$$G = Sync\Big(\{G_1, \ldots, G_n\}, \bigcup_{i=1}^{n} \Sigma_{s_i}\Big).$$

A behaviour of the system is then represented by a transition path in G starting from the initial state $x_0 = (x_{0_1}, \ldots, x_{0_n})$ of G. In the following, a *path* will always denote a transition path starting from the initial state. Moreover, as in [Sampath *et al.*, 1995], the observable behaviour of the system is supposed to be *live*, which means that for any infinite path p of G, the sequence of observable events associated with p and denoted $obs(p)$, is infinite.

The classical diagnosability property of the system is a global property defined on the paths in G [Sampath *et al.*, 1995]. Two paths p and p' in G are *distinguishable* iff $obs(p) \neq obs(p')$. Let $p = p_F s_F$ denote a path such that p_F is a path ending with the occurrence of a fault F to a state x_F and s_F is a subpath whose initial state is x_F. The diagnosability of fault F is then defined as follows:

Definition 2 *F is diagnosable iff*

$$\forall p = p_F s_F, \exists l \in \mathbb{N}, |obs(s_F)| > l \Rightarrow$$

$$(\forall p' \text{ such that } obs(p) = obs(p'), F \text{ occurs in } p')$$

If a fault is diagnosable then a diagnostic algorithm can diagnose its occurrence with certainty based on a finite sequence of observations. Diagnosability checking thus requires the search for an infinite path p', i.e. a path containing a cycle, with $obs(p) = obs(p')$ such that F is not in p'. The pair (p, p') is called a *critical pair* [Cimatti *et al.*, 2003].

2.2 Twin plant method for checking diagnosability

We now recall a centralised method to check whether F is diagnosable or not: the *twin plant* method [Jiang *et al.*, 2001]. For the sake of clarity in the rest of the paper, the *twin plant* method is presented in an original manner, based on the decentralised model instead of the global model.

Definition 3 (Local diagnoser) *The local diagnoser of a component G_i is the nondeterministic finite state machine $\widetilde{G}_i = \langle \widetilde{X}_i, \widetilde{\Sigma}_i, \widetilde{x}_{0_i}, \widetilde{T}_i \rangle$ where*

- *$\widetilde{X}_i$ is the set of states ($\widetilde{X}_i \subseteq X_i \times \mathcal{F}$ with $\mathcal{F} \subseteq 2^{\Sigma_{f_i}}$),*
- *$\widetilde{\Sigma}_i$ is the set of events ($\widetilde{\Sigma}_i = \Sigma_{o_i} \cup \Sigma_{s_i}$),*
- *$\widetilde{x}_{0_i} = (x_{0_i}, \emptyset)$ is the initial state, and*
- *$\widetilde{T}_i \subseteq \widetilde{X}_i \times \widetilde{\Sigma}_i \times \widetilde{X}_i$ is the transition set $(x, \mathcal{F}) \xrightarrow{\sigma} (x', \mathcal{F}')$ such that there exists a transition sequence $x \xrightarrow{\sigma_1} x_1 \cdots \xrightarrow{\sigma_m} x_m \xrightarrow{\sigma} x'$ in G_i with $\Sigma'_i = \{\sigma_1, \ldots, \sigma_m\} \subseteq \Sigma_{f_i} \cup \Sigma_{u_i}$ and $\mathcal{F}' = \mathcal{F} \cup (\Sigma'_i \cap \Sigma_{f_i})$.*

The top of Figure 2 depicts such a local diagnoser. The basic idea of a twin plant is to build a machine that compares every pair of paths (p, p') in the system that have the same observable behaviour, i.e. $obs(p) = obs(p')$. A *local twin plant* is essentially the same type of machine but the paths are local to a component. A local twin plant is based on the synchronisation of two instances $\widetilde{G}_i^l$ (left) and $\widetilde{G}_i^r$ (right) of the same diagnoser based on the observable events $\Sigma_{o_i} = \Sigma_{o_i}^l = \Sigma_{o_i}^r$. Since only observable behaviours are compared, the shared events must be distinguishable in both instances: in $\widetilde{G}_i^l$ (resp. $\widetilde{G}_i^r$), any shared event $\sigma \in \Sigma_{s_i}$ from $\widetilde{G}_i$ is renamed l:$\sigma \in \Sigma_{s_i}^l$ (resp. r:$\sigma \in \Sigma_{s_i}^r$).

Definition 4 (Local twin plant) *The local twin plant of G_i is the finite state machine*

$$\hat{G}_i = Sync\big(\{\widetilde{G}_i^l, \widetilde{G}_i^r\}, \Sigma_{o_i}\big).$$

One state of a local twin plant is a couple $\hat{x} = ((x^l, \mathcal{F}^l), (x^r, \mathcal{F}^r))$ that represents two possible diagnoses (the left one and the right one) given the same observable sequence. If F belongs to $\mathcal{F}^l \cup \mathcal{F}^r$ but is not in $\mathcal{F}^l \cap \mathcal{F}^r$ then F cannot be diagnosed in this state with certainty. In this case, the state $\hat{x}$ is called *F-nondiagnosable*. Otherwise it is called *F-diagnosable*. Figure 2 illustrates a local twin plant for the diagnoser shown at the top of the same figure. Its state labels (top) are composed of the state labels of $\widetilde{G}_1^l$ (middle) and $\widetilde{G}_1^r$ (bottom). Circular nodes denote f_1-nondiagnosable states.

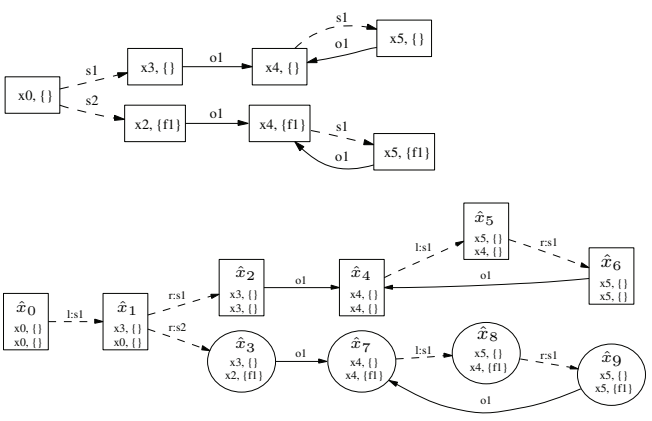

Figure 2: Local diagnoser $\widetilde{G}_1$ (top) and part of the twin plant $\hat{G}_1$ (bottom) of component G_1 shown in Figure 1.

In the following, ω represents any set of components $\mathbb{G}_\omega = \{G_{j_1}, \ldots, G_{j_{|\omega|}}\}$ of the system with $|\omega| \geq 1$. The ω-coupled twin plant is the twin plant of ω obtained by synchronisation of the local twin plants.

Definition 5 (ω-coupled twin plant) *The ω-coupled twin plant is*

$$\hat{G}_\omega = Sync\Big(\{\hat{G}_{j_1}, \ldots, \hat{G}_{j_{|\omega|}}\}, \bigcup_{i=j_1}^{j_{|\omega|}} (\Sigma_{s_i}^l \cup \Sigma_{s_i}^r)\Big).$$

By extension, a state $\hat{x} = (\hat{x}_{j_1}, \ldots, \hat{x}_{j_{|\omega|}})$ is called *F-nondiagnosable* if there exists a state $\hat{x}_{j_m}$ which is locally *F-nondiagnosable*.

The synchronisation operation $Sync$ is commutative and associative, moreover $\Sigma_{s_i}^l$, $\Sigma_{s_i}^r$ and Σ_{o_i} are disjoint sets by definition. Therefore any ω-coupled twin plant can be also obtained by first synchronising the local diagnosers over the set of shared events, to obtain the diagnoser of ω, followed by the synchronisation of two instances of the later over the set of observable events.

Consequently, considering $\omega = \mathbb{G} = \{G_1, \ldots, G_n\}$ the $\mathbb{G}$-coupled twin plant represents exactly the *global twin plant* (GTP for short) of the system where all paths of G with the same observable behaviour are compared, hence the following fundamental result.

Theorem 1 *F is diagnosable in G iff, in the $\mathbb{G}$-coupled twin plant, there is no path p with a cycle containing at least one observable event and one F-nondiagnosable state.*

Such a path p represents a critical pair (p_1, p_2): in the following, p is called a *critical path*. The twin plant method searches for such a path in the GTP.

3 Theoretical framework

We now show how to distribute the diagnosability check for a fault F without computing the GTP. The component where F may occur is denoted G_F in the following. The terms *diagnosable* and *nondiagnosable* respectively denote *F-diagnosable* and *F-nondiagnosable*. Furthermore, $Sync(\mathbb{G})$ denotes, without ambiguity, the product of all finite state machines in $\mathbb{G}$ synchronised over their shared events only.

Considering the set of local twin plants $\{\hat{G}_1, \ldots, \hat{G}_n\}$ defined in section 2, only $\hat{G}_F$ contains diagnosability information about F. The main idea of the distributed checking is to propagate firstly the diagnosability information to the other local twin plants. After the propagation, it is possible to decide locally which parts of the local twin plants are relevant to solve the diagnosability problem. Finally, for each local twin plant, a *reduced twin plant* is computed by removing all irrelevant parts.

3.1 Propagation of diagnosability information

The propagation of the diagnosability information from $\hat{G}_F$ to all other local twin plants is based on their connectivity with respect to $\hat{G}_F$.

Definition 6 (α-connectivity $Con(\alpha, \gamma)$) *The set of finite state machines connected to the finite state machine γ by distance α is recursively defined as follows.*

$Con(0, \gamma) = \{\gamma\}$

$Con(\alpha, \gamma) = \{\gamma_1 \mid \exists \gamma_2 \in Con(\alpha - 1, \gamma)$ *such that* γ_1 *and* γ_2 *share at least one event and* $\gamma_1 \notin Con(\beta, \gamma)$ *for all* $\beta < \alpha\}$.

Components γ_1 and γ_2 are *connected* if $\gamma_1 \in Con(1, \gamma_2)$. The set $transCon(\gamma)$ denotes the set of components whose behaviour can possibly influence that of γ ($transCon(\gamma) =$

$\bigcup_{\alpha=0}^{n-1} Con(\alpha,\ \gamma)$) where n is the number of components in the system.

Reducing a local twin plant $\hat{G}_i$ consists in finding a set of states, denoted $\mathbb{P}(\hat{G}_i)$, that can possibly be involved in the nondiagnosable states of GTP. These *possibly nondiagnosable states* $\mathbb{P}(\hat{G}_i)$ are determined by analysing the connectivity between the component G_i and the component G_F.

Definition 7 (Possibly nondiagnosable states) *The set of* possibly nondiagnosable states $\mathbb{P}(\hat{G}_i)$ *of a local twin plant $\hat{G}_i$ is determined as follows.*

1. $\mathbb{P}(\hat{G}_i) = \{x \in \hat{X}_i \mid x \text{ is nondiagnosable}\}$ *if* $G_i \in Con(0,\ G_F)$ *(i.e.* $G_i = G_F$*)*
2. $\mathbb{P}(\hat{G}_i) = \hat{Y}_i$ *if* $G_i \in Con(\alpha,\ G_f)$ *with* $\alpha \in \mathbb{N}^+$ *and for all states* $\hat{y} \in \hat{Y}_i$ *and all connected twin plants* $G_j \in Con(\alpha - 1,\ G_F)$ *there exists a state* $(\hat{y}_i, \hat{x}_j)$ *in the twin plant* $Sync(\{\hat{G}_i, \hat{G}_j\})$ *such that* $\hat{x}_j$ *is possibly nondiagnosable.*
3. $\mathbb{P}(\hat{G}_i) = \hat{X}_i$ *if* $G_i \notin transCon(\hat{G}_F)$

This definition can then be extended to any state $\hat{x} = (\hat{x}_{j_1}, \ldots, \hat{x}_{j_{|\omega|}})$ from any ω-coupled twin plant:

Definition 8 *A state* $\hat{x} = (\hat{x}_{j_1}, \ldots, \hat{x}_{j_{|\omega|}})$ *in a ω-coupled twin plant $\hat{G}_\omega$ is* possibly nondiagnosable *iff*

$$\forall i \in \{1, \ldots, |\omega|\}, \hat{x}_{j_i} \in \mathbb{P}(\hat{G}_{j_i}).$$

For example, to propagate the diagnosability information of f_1 in G_1 to G_2 (see Figure 1) we compute $\hat{G}_1$, the local nondiagnosable states (see Figure 2) and finally the twin plant $\hat{G}_2$ (see Figure 3). Since $\hat{G}_1$ and $\hat{G}_2$ share the events $l{:}s_1$ and $r{:}s_1$ they are connected ($\hat{G}_2 \in Con(1,\ \hat{G}_1)$). Thus to obtain $\mathbb{P}(\hat{G}_2)$, a synchronisation of $\hat{G}_1$ and $\hat{G}_2$ leading to $\hat{G}'$ is required (see Figure 3). In $\hat{G}'$ only states $(\hat{x}_3, \hat{y}_3)$ and $(\hat{x}_7, \hat{y}_3)$ are composed of a state in $\mathbb{P}(\hat{G}_1)$. Thus only state $\hat{y}_3$ could be possibly nondiagnosable. Since $\hat{G}_1$ is the only connected twin plant in $Con(1-1,\ \hat{G}_1)$ there is no need to consider other local twin plants to check whether $\hat{y}_3$ is indeed in $\mathbb{P}(\hat{G}_2)$. Therefore the set of possibly nondiagnosable states in $\hat{G}_2$ is $\mathbb{P}(\hat{G}_2) = \{\hat{y}_3\}$.

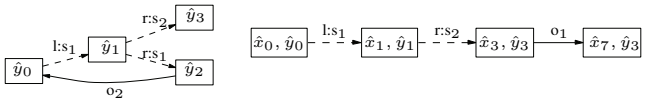

Figure 3: Part of twin plant $\hat{G}_2$ (left) for component G_2 (see Figure 1) and its synchronisation $\hat{G}'$ (right) with the part of $\hat{G}_1$ depicted in Figure 2.

Every state that is not possibly nondiagnosable is certainly diagnosable. This results from the following relationship between nondiagnosable and possibly nondiagnosable states of the GTP:

Theorem 2 *A state $\hat{x}$ in the GTP is nondiagnosable iff it is possibly nondiagnosable.*

Proof:

($\Rightarrow$) Suppose there exists a state $\hat{x} = (\hat{x}_1, \ldots, \hat{x}_n)$ in the GTP such that $\hat{x}$ is nondiagnosable and a state $\hat{x}_i$ such that $\hat{x}_i \notin \mathbb{P}(\hat{G}_i)$. Cond. 3 of def. 7 implies that $\hat{G}_i$ is in $transCon(\hat{G}_F)$. Since $\hat{x}$ is nondiagnosable the state $\hat{x}_F$ from $\hat{G}_F$, also contained in $\hat{x}$, is nondiagnosable. It follows that $\hat{x}_F \in \mathbb{P}(\hat{G}_F)$ (see cond. 1 of def. 7), so $\hat{G}_i$ is in $transCon(\hat{G}_F) \setminus \{\hat{G}_F\}$. Therefore, there exists an $\alpha > 1$ such that $\hat{G}_i \in Con(\alpha,\ \hat{G}_F)$.
Cond. 2 of def. 7 ensures that there is a twin plant $\hat{G}_j \in Con(\alpha - 1,\ \hat{G}_F)$ in which all states $\hat{x}'_j$, for which $(\hat{x}'_j, \hat{x}_i)$ is in $Sync(\{\hat{G}_j, \hat{G}_i\})$, do not belong to $\mathbb{P}(\hat{G}_j)$. Since GTP results from the synchronisation of $\hat{G}_j$ and $\hat{G}_i$, it means that $\hat{x}_i \notin \mathbb{P}(\hat{G}_i)$ implies that $\hat{x}_j \notin \mathbb{P}(\hat{G}_j)$ where $\hat{x}_j$ denotes the state from $\hat{G}_j$ contained in $\hat{x}$.
The previous reasoning led from the existence of a state $\hat{x}_i = \hat{y}_\alpha$ in a twin plant $\hat{G}_i = \hat{H}_\alpha \in Con(\alpha,\ \hat{G}_F)$ with $\hat{y}_\alpha \notin \mathbb{P}(\hat{H}_\alpha)$ to the existence of a state $\hat{x}_j = \hat{y}_{\alpha-1}$ in a twin plant $\hat{G}_j = \hat{H}_{\alpha-1} \in Con(\alpha - 1,\ \hat{G}_F)$ with $\hat{y}_{\alpha-1} \notin \mathbb{P}(\hat{G}_{\alpha-1})$. By recursively applying $\alpha - 1$ times the same reasoning, it follows that there exists a twin plant $\hat{H}_0$ belonging to $Con(0,\ \hat{G}_F)$ and a state $\hat{y}_0$ from $\hat{H}_0$ belonging to $\hat{x}$ such that $\hat{y}_0 \notin \mathbb{P}(\hat{H}_0)$. Since $\hat{G}_F$ is the only element in $Con(0,\ \hat{G}_F)$ (see def. 6) it means that $\hat{H}_0$ is actually $\hat{G}_F$ and $\hat{y}_0 = \hat{x}_F$. It finally follows that $\hat{x}_F \notin \mathbb{P}(\hat{G}_F)$, which is a contradiction.

($\Leftarrow$) Suppose there exists a state $\hat{x} \in \mathbb{P}(GTP)$ containing the local state $\hat{x}_F \in \hat{G}_F$ such that $\hat{x}$ is diagnosable. This implies that $\hat{x}_F \notin \mathbb{P}(\hat{G}_F)$. Therefore $\hat{x} \notin \mathbb{P}(GTP)$ (see def. 7) which contradicts the assumption. □

3.2 Distributed diagnosability verification

We now show how we can exploit Theorem 2 firstly to reduce the size of twin plants by removing all states that are irrelevant to the diagnosability test (see Lemma 1 below) and secondly to distribute the diagnosability test to a set of twin plants.

Definition 9 (Reduced twin plant) *A twin plant is* reduced *if all its states are either initial or on a path whose target state is possibly nondiagnosable.*

In the following, $\mathbb{R}(\hat{G})$ denotes the twin plant that is obtained by removing all states $\hat{x} \notin \mathbb{P}(\hat{G})$ of $\hat{G}$ that do not lead to a possibly nondiagnosable state. To simplify the synchronisation of reduced twin plants the event set of the latter remains unchanged ($\mathbb{R}(\hat{\Sigma}) = \hat{\Sigma}$), that is, even if $\mathbb{R}(\hat{G})$ does not contain any transitions labelled $\hat{\sigma} \in \hat{\Sigma}$, the event still belongs to its event set. Thus the following equivalence holds for any two twin plants $\hat{G}$,$\hat{G}'$: $\mathbb{R}(Sync(\{\hat{G}, \hat{G}'\})) = \mathbb{R}(Sync(\{\mathbb{R}(\hat{G}), \mathbb{R}(\hat{G}')\}))$ (see also def. 7).

Lemma 1 *The global twin plant $\hat{G} = Sync(\{\hat{G}_1, \ldots, \hat{G}_n\})$ contains exactly the same critical paths as $\mathbb{R}(\hat{G})$.*

Proof: Since the target state of every critical path p in $\hat{G}$ is part of a cycle containing at least one nondiagnosable state $\hat{x}$ (see Theorem 1), every state in p leads to $\hat{x}$ and hence to a state that is entirely composed of possibly nondiagnosable states. Thus, p is also a path in $\mathbb{R}(\hat{G})$. Further, since every path in $\mathbb{R}(\hat{G})$ is also a path in $\hat{G}$ it follows that $\mathbb{R}(\hat{G})$ contains only the critical paths defined in $\hat{G}$. □

We now show how we can decide the diagnosability problem based on a set of distributed twin plants without requiring the computation of the GTP.

Theorem 3 *A fault F in system G is diagnosable iff there exists a set of reduced twin plants $\hat{\mathbb{G}}_R = \{\hat{G}_{\omega_1}, \ldots, \hat{G}_{\omega_k}\}$ such that*

1. *$\{\omega_1, \ldots, \omega_k\}$ is a partition of the system components ($\bigcup_{i=1}^{k} \omega_i = \mathbb{G}$ and $\omega_i \cap \omega_j = \emptyset$ for all $\omega_i \neq \omega_j$) and*
2. *no plant in $\hat{\mathbb{G}}_R$ contains an observable possibly nondiagnosable cycle (OPNC for short) (i.e. a path $p = \hat{x} \xrightarrow{\hat{\sigma}} \hat{x}' \cdots \xrightarrow{\hat{\sigma}''} \hat{x}$ with at least one possibly nondiagnosable state and one observable transition label).*

Proof:

(⇒) Suppose F is diagnosable, but there does not exist such a set $\mathbb{G}^R$. The GTP, which satisfies cond. 1, also contains an OPNC and thus an observable nondiagnosable cycle (see Theorem 2). This implies that F is not diagnosable (see Theorem 1) and thus contradicts the assumption.

(⇐) Suppose now that there exists such a set $\mathbb{G}^R$ and F is not diagnosable. From the nondiagnosability of F it follows that there exists an observable nondiagnosable cycle in the GTP and hence an OPNC with an observable event $\sigma \in \Sigma_{o_i}$ (see Theorem 2). Based on the synchronisation operation (see page 2) it means that every ω-coupled twin plant $\hat{G}_\omega$ with $G_i \in \omega$ contains an OPNC. Thus every set satisfying cond. 1, and therefore involving G_i, does not satisfy cond. 2. Hence there does not exists such a set $\mathbb{G}^R$ which contradicts the assumption. □

4 Algorithm

This section presents a scalable algorithm that exploits Theorem 3 to reduce the search space of the diagnosability analysis by distributing it on several reduced twin plants. Every set of reduced twin plants $\mathbb{G}^R$ represents a partition of the components of the system and the challenge is to find one that contains no OPNCs and thus proves that F is diagnosable. OPNCs can be removed by synchronising connected twin plants. Therefore this operation is the basis for the distributed diagnosability check (see Algorithm 1).

The algorithm performs as follows. After the initialisation and the computation of the local reduced twin plants (line 3-5), the basic idea is to synchronise them pairwise using the function $Sync(\mathbb{R}(\hat{G}^i), \mathbb{R}(\hat{G}^{i'}))$. The synchronisation is performed only if either $\mathbb{R}(\hat{G}^i)$ or $\mathbb{R}(\hat{G}^{i'})$ has an OPNC. This pairwise synchronisation and the reduction of the resulting twin plant is computed using function $RedPairwiseSync(\hat{\mathbb{G}})$ (line 7). The selection of pairs to synchronise is important since it has a strong impact on the algorithm efficiency. Among the different selection heuristics, e.g. [Lamperti and Zanella, 2003], [Pencolé and Cordier, 2005], a common rule must be respected: a pair of twin plants to synchronise must be connected otherwise the synchronisation will not remove OPNCs.

The synchronisation operation is repeated step by step until the algorithm stops due to one of the following causes.

(1) None of the twin plants in $\hat{\mathbb{G}}$ contains an OPNC, that is, a partition $\mathbb{G}^R$ has then been determined which verifies that fault F is diagnosable (see Theorem 3).

(2) A twin plant contains an OPNC that cannot be removed by further synchronisations. This is the case if the reduced GTP has been computed which involves all components.

(3) Due to finite computational resources, the algorithm must stop. In this case it returns a set of twin plants with OPNCs that provides all possible reasons of why F might not be diagnosable. At this level, the nondiagnosability of F is not certain but it can be deduced that any subset of components involved in an element of the partition is not sufficient to diagnose F with certainty.

Algorithm 1 Distributed diagnosability check

```
1:  INPUT:
    - component models G_1, ... G_n of the system
    - fault F that can occur in component G_F
2:  Initialisations:
    Ĝ ← ∅                                   Set of relevant twin plants
3:  Compute local twin plants Ĝ_i for all components G_i and
    add them to Ĝ
4:  Ĝ ← GetNonDiagStateLab(Ĝ, Ĝ_F)
5:  Reduce all twin plants in Ĝ
6:  while ExistsOPNC(Ĝ), and SufficientMemory(Ĝ) and
    NrElements(Ĝ) > 1 do
7:      Ĝ ← RedPairwiseSync(Ĝ)
8:  end while
9:  if ExistsOPNC(Ĝ) is false then
10:     return "F is diagnosable"
11: else
12:     Ĝ^cyc ← GetTPsWithOPNC(Ĝ)
13:     return Ĝ^cyc
14: end if
```

In case (2), that is if F is not diagnosable, all critical paths are returned. They provide a complete diagnosability analysis of the undistinguishable behaviours that cause the nondiagnosability of F. This information also contains a *synthetic view* of the nondiagnosability of F.

Extracting this synthetic view from the OPNCs consists in considering every nondiagnosable state $\hat{x} = (\hat{x}_{j_1}, \ldots, \hat{x}_{j_{|\omega|}}) = ((x^l_{j_1}, \mathcal{F}^l_{j_1}), (x^r_{j_1}, \mathcal{F}^r_{j_1})), \ldots,$

$((x^l_{j_{|\omega|}}, \mathcal{F}^l_{j_{|\omega|}}), (x^r_{j_{|\omega|}}, \mathcal{F}^r_{j_{|\omega|}}))$ in the OPNCs. If one label contains F (i.e. $\mathcal{F}^l = \bigcup_{i=1}^{|\omega|} \mathcal{F}^l_i$) and the other label is empty (i.e. $\mathcal{F}^r = \bigcup_{i=1}^{|\omega|} \mathcal{F}^r_i = \emptyset$) it means that there exists an infinite nominal behaviour in the system which cannot be distinguished from a faulty behaviour. In this case, the observation of the system does not guarantee that a fault can be *detected*, it must be updated (respecified) in order to guarantee the diagnosability of F. If the label $\mathcal{F}^r$ is not empty, then the diagnosability problem results from the fact that the occurrence of the faults $\mathcal{F}^r$ cannot be discriminated from the occurrence of F. In this case, a designer can decide that the discrimination of F with $\mathcal{F}^r$ is not important and abstract the set of faults to only diagnose a fault $F_{abs} = F \vee \mathcal{F}^r$.

5 Discussion

Diagnosability on DES is a problem that has been introduced in [Sampath *et al.*, 1995] where the authors solve it by detecting some transition cycles of ambiguous states in a *global diagnoser*. The global diagnoser is equivalent to the determination of the local diagnoser (see Definition 3) where the global model G is seen as a component. The main drawback of this method results from the diagnoser computation which is exponential in the number of states in the global model (determination) and as a consequence is doubly exponential in the number of components in the system. [Jiang *et al.*, 2001; Yoo and Lafortune, 2002] then propose new algorithms which are only polynomial in the number of states in G and which introduce the twin plant method. The question of efficiency is raised in [Cimatti *et al.*, 2003] where the authors propose to use symbolic model-checking to test a restrictive diagnosability property by taking advantages of efficient model-checking tools. But, still the diagnosability problem is seen as a test on a system and not as a deep analysis of the reasons why a system is not diagnosable. In [Pencolé, 2004], the author introduces the problem of analysing diagnosability of a system in a decentralised way by aggregating local non reduced twin plants until a critical path is detected. In this method, the diagnosability analysis is not scalable and only partial since the feedback is just about the detected critical path.

6 Conclusion and future work

Our theoretical framework solves the diagnosability problem in a distributed manner and takes into account the distributed nature of complex event-driven systems. We have also proposed an algorithm that is scalable in the sense that, if the problem is too complex to be solved with available computational resources, it can still provide an approximate analysis of the diagnosability of the system. In case of a negative answer to the diagnosability problem, we claim that finding a counter example is not sufficient to assist any designer of a system to improve diagnosability. We thus propose a synthetic view of the diagnosability results that summarise the different reasons why a fault is not diagnosable.

The perspectives of this work are numerous: it is a clear first step to assist any designer to make diagnosable systems because of the synthetic view of the results instead of a single counter example. One of our next challenge is the application of this framework to provide design recommendations for improving the diagnosability of health monitoring system for aircraft maintenance [Ghelam *et al.*, 2006] or to provide an assistance to the design of composite Web Services [Yan *et al.*, 2005]. Another challenge is to improve our algorithm in order to search for *optimal partitions* where each element of the partition is as small as possible. From an optimal partition, we can then derive optimal diagnostic algorithms based on set of small system parts that still guarantee an accurate global diagnosis.

Acknowledgments

The authors thank Jussi Rintanen, Alban Grastien and the anonymous reviewers for useful discussions and comments.

References

[Cimatti *et al.*, 2003] A. Cimatti, C. Pecheur, and R. Cavada. Formal verification of diagnosability via symbolic model checking. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence IJCAI'03*, pages 363–369, Acapulco, Mexico, 2003.

[Ghelam *et al.*, 2006] S. Ghelam, Z. Simeu-Abazi, J.-P. Derain, C. Feuillebois, S. Vallet, and M. Glade. Integration of health monitoring in the avionics maintenance systems. In *Proceedings of SAFEPROCESS'06* , pages 1519–1524, Beijing, China PR, 2006.

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

[Lamperti and Zanella, 2003] G. Lamperti and M. Zanella. *Diagnosis of active systems*. Kluwer Academic Publishers, 2003.

[Pencolé and Cordier, 2005] Y. Pencolé and M.-O. Cordier. A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks. *Artificial Intelligence*, 164:121–170, May 2005.

[Pencolé, 2004] Y. Pencolé. Diagnosability analysis of distributed discrete event systems. In *European Conference on Artificial Intelligence (ECAI'04)*, pages 43–47, Valencia, Spain, 2004.

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

[Yan *et al.*, 2005] Y. Yan, Y. Pencolé, M.-O. Cordier, and A. Grastien. Monitoring web service networks in a model-based approach, 3rd european conference on web services (ecows05). In *3rd European Conference on Web Services (ECOWS05)*, Växjö, Sweden, November 2005.

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