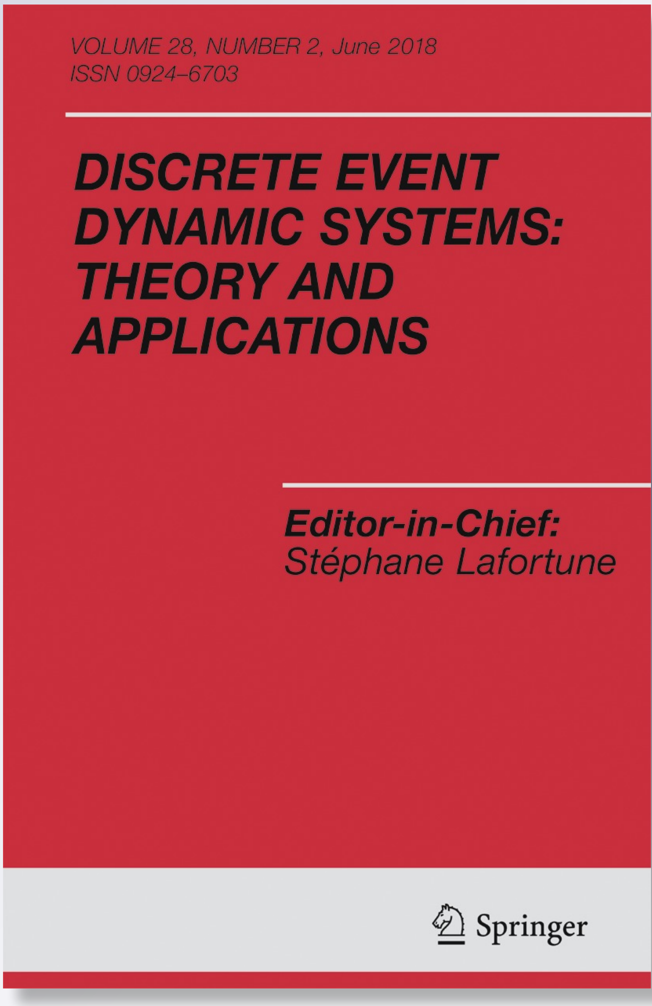
Codiagnosability of networked discrete event systems subject to communication delays and intermittent loss of observation

Carlos E. V. Nunes, Marcos V. Moreira, Marcos V. S. Alves, Lilian K. Carvalho & João Carlos Basilio

Discrete Event Dynamic Systems
Theory and Applications

ISSN 0924-6703
Volume 28
Number 2

Discrete Event Dyn Syst (2018)
28:215-246
DOI 10.1007/s10626-017-0265-6



Springer

Discrete Event Dyn Syst (2018) 28:215–246
https://doi.org/10.1007/s10626-017-0265-6

Codiagnosability of networked discrete event systems subject to communication delays and intermittent loss of observation

Carlos E. V. Nunes[1] · Marcos V. Moreira[1] · Marcos V. S. Alves[1] · Lilian K. Carvalho[1] · João Carlos Basilio[1]

Received: 24 September 2016 / Accepted: 27 December 2017 / Published online: 20 January 2018

Abstract Failure diagnosis is a crucial task in modern industrial systems, and several works in the literature address this problem by modeling the system as a Discrete-Event System (DES). Most of them assume perfect communication between sensors and diagnosers, *i.e.*, no loss of observation of events, or event communication delays between the measurement sites and the diagnosers. However, industrial systems can be large and physically distributed, in which cases, communication networks are used to provide an efficient way to establish communication between devices. In diagnosis systems, the use of networks can introduce delays in the communication of event occurrences from measurement sites to the local diagnosers, leading to an incorrect observation of the order of occurrence of events generated by the system and, as a consequence, to an incorrect diagnosis decision by the local diagnoser. In this paper, we address the problem of decentralized diagnosis of networked

The work of M. V. Moreira was partially supported by the Brazilian Research Council (CNPq), grants number 309084/2014-8 and 200536/2017-6.

This article belongs to the Topical Collection: *Special Issue on Diagnosis, Opacity and Supervisory Control of Discrete Event Systems*
Guest Editors: Christos G. Cassandras and Alessandro Giua

✉ João Carlos Basilio
basilio@dee.ufrj.br

Carlos E. V. Nunes
carlosevnunes@poli.ufrj.br

Marcos V. Moreira
moreira.mv@poli.ufrj.br

Marcos V. S. Alves
mvalves@poli.ufrj.br

Lilian K. Carvalho
lilian@dee.ufrj.br

[1] Department of Electrical Engineering, Universidade Federal do Rio de Janeiro, 21949-900, Rio de Janeiro, R.J, Brazil

Discrete-Event Systems subject to event communication delays, and we introduce the definition of network codiagnosability of the language generated by a DES subject to both event communication delays and intermittent loss of observation, and present necessary and sufficient conditions for a language to be network codiagnosable, for short. We also propose an algorithm to verify this property.

Keywords Language codiagnosability · Automaton · Networked discrete-event systems · Communication delays

1 Introduction

Modern industrial plants can be large and physically distributed, with several devices exchanging information among them. In these cases, the conventional structure of dedicated point-to-point communication is complex, expensive, difficult to maintain due to the large quantity of cables and connectors (Huo et al. 2004), and, in some cases, impossible to be implemented in a real system. In order to reduce the costs of implementation and maintenance, and also to provide an efficient way to establish communication between several devices in an industrial system, communication networks are used.

In diagnosis systems based on communication networks, the intense data traffic in communication channels, or the long distance between measurement sites and diagnosers, can delay the reception by a local diagnoser of the information communicated through the channel. Moreover, measurement sites can route their messages to several diagnosers, which can also delay the reception of the information by the local diagnosers, and, consequently, the diagnoser receives the information about the occurrence of the events in an order different from the order the events have been transmitted by the different measurement sites (Debouk et al. 2003; Park and Cho 2006). Other factors, such as sensor faults and communication channel problems, may prevent a signal issued by a sensor from reaching the local diagnosers. In both cases, the diagnoser can either make a wrong decision regarding a failure occurrence, or it can observe an event that is not feasible in its current state and gets stuck. The problem of delay in communication networks has been addressed in Debouk et al. (2003) and Qiu and Kumar (2008) for fault diagnosis of DES, and the problem of discrete-event systems subject to unreliable observations of events has been addressed in Athanasopoulou et al. (2010); Carvalho et al. (2011, 2012, 2013); Takai (2012).

In this paper, we address the problem of failure diagnosis of networked DES with the decentralized diagnosis structure proposed in Protocol 3 of Debouk et al. (2000), *i.e.*: (i) there is no communication between local diagnosers; (ii) each local diagnoser infers the occurrence of the failure event based on its own observations; (iii) the failure event is diagnosed when at least one of the local diagnosers identifies its occurrence. We also consider that the observation of event occurrences is distributed in the plant, *i.e.*, the plant has several measurement sites, and each site has exclusive communication channels to send the information regarding event occurrences to local networked diagnosers, as shown in Fig. 1. In addition, we assume the existence of communication delays between measurement sites and local networked diagnosers, which may result in an observation order different from the actual order of event occurrences in the plant.

The problem addressed in this paper is different from the diagnosis problems of networked systems proposed in the literature. The problem of decentralized failure diagnosis subject to communication delays between local diagnosers and the coordinator, under

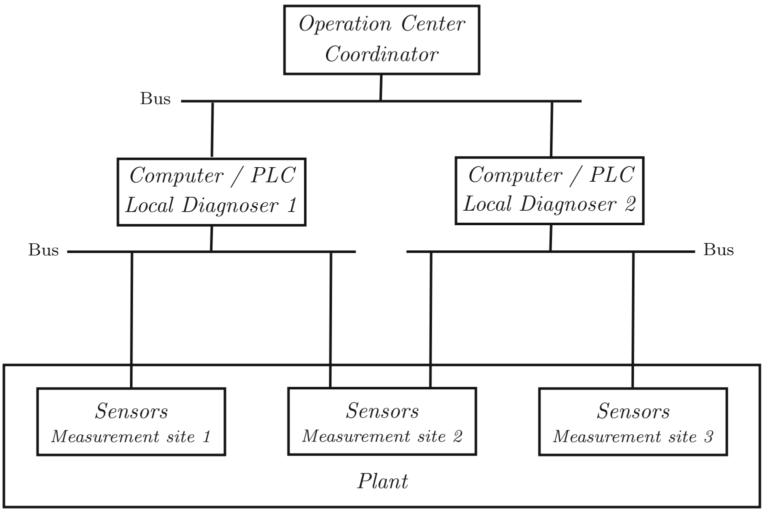


Fig. 1 Networked decentralized diagnosis scheme

Protocols 1 and 2 of Debouk et al. (2000), is proposed in Debouk et al. (2003). In Debouk et al. (2003), it is assumed that the events received by the coordinator can be observed out of their original order of occurrence; however, no delay between the measurement sites and the diagnoser is assumed in Debouk et al. (2003). Here we consider Protocol 3 of Debouk et al. (2000) and assume communication delays between the measurement sites and the local diagnosers. Finite delays in the communication between local diagnosers and coordinator are not assumed here since they do not affect the diagnosis decision. The problem proposed in this paper is also different from the so-called distributed diagnosis scheme proposed in Qiu and Kumar (2008), where each local diagnoser can exchange information with the other local diagnosers to infer the failure event occurrence. In addition, in Qiu and Kumar (2008) the communication delay between two local diagnosers is considered equal, and it is assumed that there is no delay between the measurement sites and diagnosers.

It is important to remark that the problem of communication delays has also been addressed in the context of supervisory control of DES by Balemi (1994); Park and Cho (2006, 2007b); Lin (2014); Shu and Lin (2015), for the monolithic case, and by Park and Cho (2007a) and Shu and Lin (2014), for the decentralized/distributed case. In the aforementioned works, it is assumed that there is only one communication channel between the plant and supervisor, and, thus, no change in the order of event observations by the supervisor occurs. Since codiagnosability is not time critical, *i.e.*, the diagnoser can detect the fault after an arbitrarily large number of event occurrences, bounded communication delays that cannot change the order of event observation are not important in the context of failure diagnosis. We consider here decentralized diagnosis of networked DES assuming that communication delays can be large enough that it can modify the order of observation of the events received by the local diagnosers. Still in the context of supervisory control, Tripakis (2004) and Sadid et al. (2015) assume that communication delays may change the order of event observation. One important restriction of these approaches is that the same delay upper bound is assumed for all communication channels. In addition, Sadid et al. (2015) restricts the problem to those systems whose automaton models have no loops of communication events (events that are subject to communication delays) in the original system. None of these assumptions are assumed here. Figure 2 shows the main differences between our approach and others previously presented in the literature regarding the location of the communication channels subject to delays, and the number of communication channels/effect of communication delays. Not directly related to our work, we cite the works by Rohloff (2005), Sánchez and Montoya (2006), Lin (2014), Alves et al. (2014), and Ushio and Takai (2016), that consider the problem of loss of observations (permanent or intermittent), in the context of supervisory control.

In this paper, we first introduce the definition of network codiagnosability with respect to event communication delays and intermittent loss of observation, to be referred here simply to as network codiagnosability, and then, we propose an algorithm to construct deterministic automata that capture the effect of event communication delays in the communication channels between the measurement sites and the local diagnosers. The problem of intermittent loss of observation is addressed by using the dilation function proposed in Carvalho et al. (2012). Based on the model of the system obtained to represent the effect of the communication delays and intermittent loss of observation, we present a necessary and sufficient condition for network codiagnosability, and develop an algorithm for its verification.

This paper is organized as follows. In Section 2 we present preliminary concepts on DES necessary in the sections that follow. In Section 3 we formulate the problem of decentralized diagnosis of systems with network communication subject to event delays and loss of observation, and, in the sequel, we present an algorithm to obtain automata that model all possible delays in the communication of events to local diagnosers. The problem of intermittent loss of observation is considered in the sequel, by using the dilation function in the model of the system with communication delays. The definition of network codiagnosability is also presented in Section 3. In Section 4 we present an algorithm to verify the network codiagnosability of DESs. In Section 5 we analyze the computational complexity of the

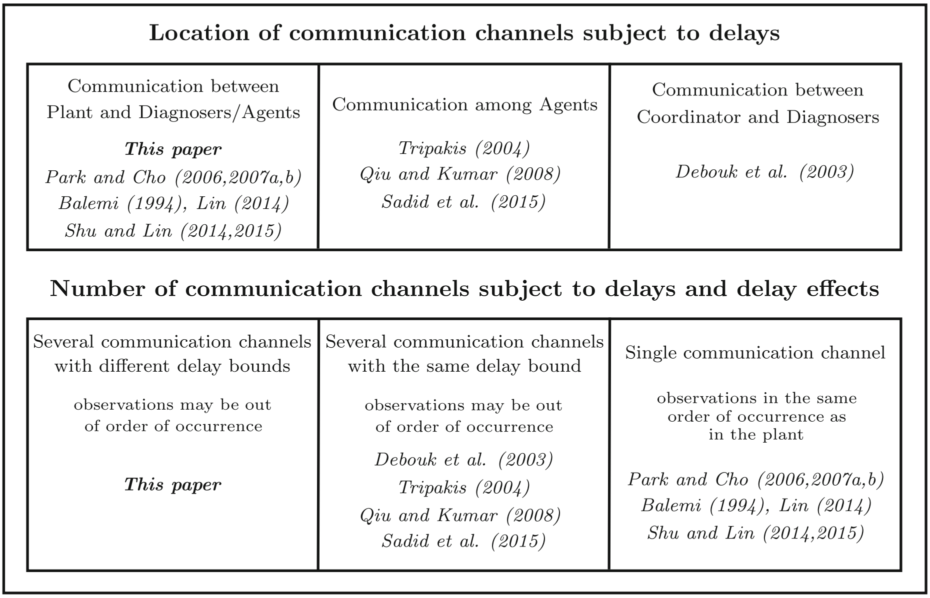


Fig. 2 Comparison among different networked DES regarding the location of the communication channels subject to delays, and the number of communication channels/effect of communication delays

algorithm for the verification of the network codiagnosability. Finally, in Section 6 we list the main contributions of the paper. A running example is used to illustrate the main results of the paper.

A preliminary version of this paper was presented at WODES2016 (Nunes et al. 2016). Here, besides presenting the proofs of the theorems stated in the conference paper, we also consider intermittent loss of observation (Section 3.3) and make appropriate changes in the sections that follow.

2 Preliminaries

2.1 Definitions and notations

Let $G = (X, \Sigma, f, \Gamma, x_0, X_m)$ be a deterministic automaton that models a DES, where X is the state space, Σ is the set of events, $\Gamma : X \to 2^{\Sigma}$ is the active event function, $f : X \times \Sigma^* \to X$ is the state transition function partially defined in $X \times \Sigma^*$, where Σ^* denotes the Kleene closure of Σ, X_m is the set of marked states, and x_0 is the initial state (Cassandras and Lafortune 2008). Assume that event set Σ is partitioned as $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, where Σ_o and Σ_{uo} denote, respectively, the observable and unobservable event sets. The language generated by G will be denoted by $L(G)$ or simply by L_G.

Let K be a language defined over Σ^*. The prefix closure of K is denoted by $\overline{K}$. If $K = \overline{K}$, then K is said to be prefix-closed. A language $M \subseteq \Sigma^*$ is said to be live if, for all traces $s \in M$, there exists an event $\sigma \in \Sigma$, such that $s\sigma \in M$. In this regard, the language generated by G is live if $\Gamma(x) \neq \emptyset$, for all $x \in X$.

The accessible part of G, denoted as $Ac(G)$, is the automaton obtained by deleting all states of G, and their related transitions, that are not reachable from the initial state x_0. The coaccessible part of G, denoted as $CoAc(G)$, is the automaton obtained by deleting all states of G from which it is not possible to reach a marked state.

The projection $P_{ls} : \Sigma_l^* \to \Sigma_s^*$, where $\Sigma_s \subset \Sigma_l$ is defined as: (i) $P_{ls}(\epsilon) = \epsilon$; (ii) $P_{ls}(\sigma) = \sigma$ if $\sigma \in \Sigma_s$; (iii) $P_{ls}(\sigma) = \epsilon$, if $\sigma \in \Sigma_l \setminus \Sigma_s$; (iv) $P_{ls}(t\sigma) = P_{ls}(t)P_{ls}(\sigma)$, for $t \in \Sigma_l^*$ and $\sigma \in \Sigma_l$, where ϵ denotes the empty trace. The inverse projection $P_{ls}^{-1} : \Sigma_s^* \to 2^{\Sigma_l^*}$ is defined as $P_{ls}^{-1}(q) = \{t \in \Sigma_l^* : P_{ls}(t) = q\}$.

Let G_1 and G_2 be two automata. The parallel composition between G_1 and G_2 is denoted by $G_1 \| G_2$, and is defined as: $G_1 \| G_2 = Ac(X_1 \times X_2, \Sigma_1 \cup \Sigma_2, f_{1\|2}, \Gamma_{1\|2}, (x_{0,1}, x_{0,2}), X_{m_1} \times X_{m_2})$, where $f_{1\|2}((x_1, x_2), \sigma) = (f_1(x_1, \sigma), f_2(x_2, \sigma))$, if $\sigma \in \Gamma_1(x_1) \cap \Gamma_2(x_2)$, $f_{1\|2}((x_1, x_2), \sigma) = (f_1(x_1, \sigma), x_2)$, if $\sigma \in \Gamma_1(x_1) \setminus \Sigma_2$, $f_{1\|2}((x_1, x_2), \sigma) = (x_1, f_2(x_2, \sigma))$, if $\sigma \in \Gamma_2(x_2) \setminus \Sigma_1$, or, undefined, otherwise.

The observer $Obs(G, \Sigma_o)$ is defined as $Obs(G, \Sigma_o) = (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0_{obs}}, X_{m_{obs}})$, where $X_{obs} \subseteq 2^X$ and $X_{m_{obs}} = \{B \in X_{obs} : B \cap X_m \neq \emptyset\}$. In order to define $x_{0_{obs}}$, Γ_{obs} and f_{obs}, it is necessary to introduce the definition of unobservable reach of a state $x \in X$, denoted as $UR(x, \Sigma_o)$

$$UR(x, \Sigma_o) = \{y \in X : (\exists t \in \Sigma_{uo}^*)[f(x, t) = y]\}.$$

The unobservable reach can be extended to a set $B \in 2^X$ as:

$$UR(B, \Sigma_o) = \bigcup_{x \in B} UR(x, \Sigma_o).$$

Thus, $x_{0,obs} = UR(x_0, \Sigma_o)$, and for all $x_{obs} \in X_{obs}$, $\Gamma_{obs}(x_{obs}) = \bigcup_{x \in x_{obs}} \Gamma(x)$, $f_{obs}(x_{obs}, \sigma) = \bigcup_{(x \in x_{obs}) \wedge (f(x,\sigma)!)} UR[f(x, \sigma), \Sigma_o]$, if $\sigma \in \Gamma_{obs}(x_{o,obs})$, or, undefined, otherwise, $f(x, \sigma)!$ denotes that $f(x, \sigma)$ is defined, *i.e.*, $\exists y \in X$ such that $f(x, \sigma) = y$.

Let $\Sigma_o = \Sigma_{ilo} \dot{\cup} \Sigma_{nilo}$ be a partition of Σ, where Σ_{ilo} is the set of observable events subject to intermittent loss of observations and Σ_{nilo} is the set of observable events that are not subject to intermittent loss of observation. In addition, let $\Sigma'_{ilo} = \{\sigma' : \sigma \in \Sigma_{ilo}\}$ be a set of unobservable events, and $\Sigma_{dil} = \Sigma \cup \Sigma'_{ilo}$. The dilation function is defined as $D : \Sigma^* \to 2^{(\Sigma_{dil})^*}$, where, $D(\epsilon) = \{\epsilon\}$, $D(\sigma) = \{\sigma\}$, if $\sigma \in \Sigma \setminus \Sigma_{ilo}$, $D(\sigma) = \{\sigma, \sigma'\}$, if $\sigma \in \Sigma_{ilo}$ and $D(s\sigma) = D(s)D(\sigma)$ where $s \in \Sigma^*$ and $\sigma \in \Sigma$. The dilation operation D can be extended to languages as follows: $D(L) = \bigcup_{s \in L} D(s)$. The reader is referred to Carvalho et al. (2012) for more insights into the definition of dilation operation.

Let $\sigma \in \Sigma$ and $s \in \Sigma^*$. Then, with a slight abuse of notation, $\sigma \in s$ denotes that event σ is one of the events that form trace s, and $\sigma^{(l)}$ denotes the l-th occurrence of event $\sigma \in s$, that is, $\sigma^{(l)} \in s$ implies that there are, at least, l occurrences of event σ in trace s.

2.2 Codiagnosability of discrete event systems

Let $\Sigma_f \subseteq \Sigma_{uo}$ be the set of failure events. For the sake of simplicity, and without loss of generality, we assume in this paper that there is only one failure event, *i.e.*, $\Sigma_f = \{\sigma_f\}$.

Definition 1 *(normal and failure traces)*: Let $s \in L_G$, and define $L_s = \{s\}$. Then s is a failure trace if $\exists s_p \in \overline{L_s}$ such that $s_p = \tilde{s}_p \sigma_f$ for some $\tilde{s}_p \in \Sigma^*$. Otherwise, s is a normal trace.

According to Definition 1, a failure trace is a sequence of events s such that σ_f is one of its events and a normal trace, on the other hand, does not contain the event σ_f.

The set of all normal traces generated by the system is the prefix-closed language $L_N \subset L_G$. Thus, the set of all failure traces is given by $L_G \setminus L_N$.

Let G_N be the subautomaton of G that models the normal language of the system with respect to the failure event set Σ_f. Then, the language generated by G_N is L_N.

In this paper, we adopt the decentralized diagnosis scheme presented in Protocol 3 of Debouk et al. (2000), that consists of a set of n local diagnosers that do not communicate with each other. In addition, each local diagnoser infers the occurrence of the failure event based on its own set of observable events $\Sigma_{o_i} \subset \Sigma_o$, where $i = 1, 2, \ldots, n$, *i.e.*, the set of events is partitioned, for each local diagnoser, as $\Sigma = \Sigma_{o_i} \dot{\cup} \Sigma_{uo_i}$, where Σ_{uo_i} denotes the set of events that are unobservable by the i-th local diagnoser. In this architecture, each local diagnoser is not capable of distinguishing all failure traces of the system from normal ones; thus it is necessary that all local diagnosers cooperate with each other in order to diagnose the occurrence of the failure event. A failure event is diagnosed when at least one of the local diagnosers identifies its occurrence. This notion of decentralized diagnosability is referred to in the literature as codiagnosability (Qiu and Kumar 2006). The definition of codiagnosability of a language L_G is as follows.

Definition 2 (Debouk et al. 2000) Let L_G and $L_N \subset L_G$ be prefix-closed languages generated by G and G_N, respectively, and let $P_{o_i} : \Sigma^* \to \Sigma^*_{o_i}$, $i = 1, \ldots, n$, be projection operations. Then, L_G is codiagnosable with respect to projections P_{o_i} and Σ_f if

$$(\exists z \in \mathbb{N})(\forall s \in L_G \setminus L_N)(\forall st \in L_G \setminus L_N, ||t|| \geq z) \Rightarrow$$
$$(\exists i \in \{1, 2, \ldots, n\})(P_{o_i}(st) \neq P_{o_i}(\omega), \forall \omega \in L_N)$$

where $||.||$ denotes the length of a trace.

Remark 1 Notice that, when $n = 1$, Definition 2 is equal to the definition of language diagnosability (Sampath et al. 1995).

According to Definition 2, L_G is codiagnosable with respect to P_{o_i} and Σ_f if, and only if, for all failure traces $s_F = st$ of arbitrarily long length after the occurrence of the failure event, there do not exist traces $s_{N_i} \in L_N$, where s_{N_j} is not necessarily different from s_{N_k} for $j \neq k$, such that $P_{o_i}(s_{N_i}) = P_{o_i}(s_F)$, for all $i \in \{1, 2, \ldots, n\}$.

2.3 Codiagnosability verification of DES

The codiagnosability verification of L_G is the first step for the failure decentralized diagnosis of a DES, and several works in the literature address this problem (Debouk et al. 2000; Qiu and Kumar 2006; Moreira et al. 2011, 2016). In this work, we use the algorithm proposed by Moreira et al. (2011) as the basis for the construction of a verifier automaton G_V for network codiagnosability verification.

3 Network codiagnosability of discrete-event systems

3.1 Problem formulation

In general, different sensors in distributed systems do not share the same communication channel. This is so because, either the measurement sites are far away from each other, or a single communication channel may not have enough capacity to transmit all data from a measurement site to a local diagnoser. Thus, the implementation of several communication channels between measurement sites and diagnosers is, in general, necessary in network-controlled systems.

In this paper, we introduce a network decentralized diagnosis scheme for a plant with different measurement sites MS_j, $j = 1, \ldots, m$, where each measurement site MS_j reads the signals associated with a subset $\Sigma_{MS_j} \subset \Sigma_o$ of the observable events of the system. In this scheme, events of Σ_{MS_j} are communicated to a local diagnoser LD_i, $i = 1, 2, \ldots, n$, by an exclusive communication channel ch_{ij}, *i.e.*, only the events detected by measurement site MS_j can be communicated through channel ch_{ij} between measurement site MS_j and local diagnoser LD_i. Let us denote the set of events communicated to local diagnoser LD_i, through communication channel ch_{ij}, as $\Sigma_{o_{ij}} \subseteq \Sigma_{MS_j}$. It is important to remark that if the communication channel ch_{yx}, between a measurement site MS_x and a local diagnoser LD_y, does not exist, then $\Sigma_{o_{yx}} = \emptyset$. Thus, the set of observable events of LD_i, Σ_{o_i}, is given by:

$$\Sigma_{o_i} = \bigcup_{j=1}^{m} \Sigma_{o_{ij}}. \tag{1}$$

It is important to notice that $\Sigma_o = \bigcup_{i=1}^{n} \Sigma_{o_i}$.

In Fig. 3, we show the network decentralized diagnosis scheme proposed in this paper for a plant with distributed observation with four measurement sites and two local diagnosers. Notice that measurement site MS_1 is capable of communicating to local diagnoser LD_1 through channel ch_{11} only the events in $\Sigma_{o_{11}} \subseteq \Sigma_{MS_1}$, and that measurement site MS_3 communicates the events in $\Sigma_{o_{13}} \subseteq \Sigma_{MS_3}$ and $\Sigma_{o_{23}} \subseteq \Sigma_{MS_3}$ to local diagnosers LD_1 and

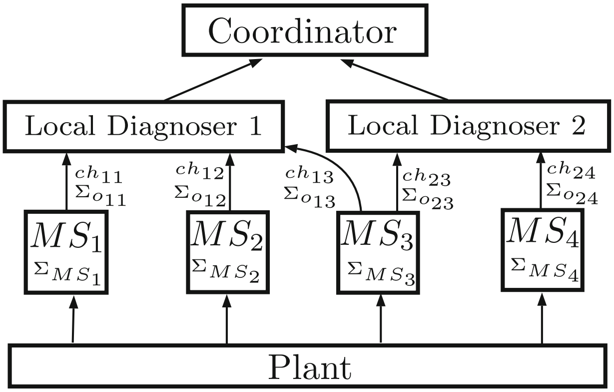


Fig. 3 Network decentralized diagnosis architecture

LD_2, respectively, through communication channels ch_{13} and ch_{23}. It is important to remark that in the network architecture proposed in this paper, a measurement site can transmit a different set of observable events to different local diagnosers, which implies that, in the example depicted in Fig. 3, $\Sigma_{o_{13}}$ can be different from $\Sigma_{o_{23}}$.

The communication between measurement sites and local diagnosers through a communication network can introduce two problems for the failure diagnosis as follows: (i) loss of data transmitted through communication channels; and (ii) delay in the communication of an event occurrence to a local diagnoser. When either one of the situations above occurs, the diagnoser may send a wrong diagnosis decision to the coordinator, and then, the implemented diagnosis scheme is no longer reliable.

Regarding event communication delays, we make the following assumptions:

A1. The delay in the communication of an event $\sigma \in \Sigma_o$ is measured by steps (Tripakis 2004), where one step is the occurrence of an event, *i.e.*, the delay is measured by the number of events that are executed by the plant after the occurrence of σ and before its observation by a local diagnoser.

A2. The event communication delays are bounded.

A3. The communication channels follow first-in first-out (FIFO) rule as far as sending and reception of events are concerned.

A4. There is one and only one channel ch_{ij} between measurement site MS_j and local diagnoser LD_i, and the maximum communication delay of channel ch_{ij}, denoted by k_{ij}, is previously known. If a channel ch_{yx} does not exist, then by convention, $k_{yx} = 0$.

A5. The event sets Σ_{MS_i} and Σ_{MS_j} are disjoint for all $i, j \in \{1, 2, \ldots, m\}$, $i \neq j$.

Regarding loss of data in communication channels, we make the following assumption:

A6. The loss of observation of events occurs in the communication channels that connect measurement sites and local diagnosers.

Therefore, according to assumption **A6**, the loss of observation of an event does not change the plant behavior, but only the observation.

3.2 Model of the plant subject to communication delays

In the system structure shown in Fig. 3, a datum transmitted by a communication channel, ch_{ip}, can delay with respect to another communication channel ch_{iq}, where $p \neq q$ and $p, q \in \{1, 2, \ldots, j\}$. As a consequence, in the communication of the events to the local diagnoser LD_i, they can be observed in an order different from their actual occurrence in the system. Thus, in order to address the problem of failure diagnosis in networked DES with communication delays, it is necessary to construct automata G_i, $i = 1, 2, \ldots, n$, that represent all possible ordering of observation of the traces executed by the plant by the local diagnosers LD_i.

To distinguish an event $\sigma \in \Sigma_{o_{ij}}$ that occurs in the plant, from its observation by local diagnoser LD_i, we create an event σ_{s_i} that represents the successful observation of σ by local diagnoser LD_i. In this regard, let

$$\Sigma^s_{o_{ij}} = \{\sigma_{s_i} : \sigma \in \Sigma_{o_{ij}}\} \tag{2}$$

denote the set of events that are observable to local diagnoser LD_i and whose occurrence are recorded at MS_j, and let

$$\Sigma^s_{o_i} = \bigcup_{j=1}^{m} \Sigma^s_{o_{ij}} \tag{3}$$

denote the set of observable events that are successfully communicated to local diagnoser LD_i. Then, the following sets of events can be defined

$$\Sigma_i = \Sigma \cup \Sigma^s_{o_i}, \ \ i = 1, \ldots, n, \tag{4}$$

where the events in Σ are now unobservable for all local diagnosers LD_i, $i = 1, \ldots, n$, and the events in $\Sigma^s_{o_i}$ are observable for local diagnoser LD_i.

The following example illustrates the observation of a trace in L_G by a local diagnoser in the presence of communication delays.

Example 1 Consider the network decentralized diagnosis scheme depicted in Fig. 4a, which consists of two local diagnosers, LD_1 and LD_2, and three measurement sites, MS_1, MS_2 and MS_3. The plant with distributed observation is modeled by automaton G depicted in Fig. 4b, where $\Sigma = \{a, b, c, d, e, \sigma_f\}$. Let $\Sigma_{MS_1} = \{a\}$, $\Sigma_{MS_2} = \{c\}$ and $\Sigma_{MS_3} = \{b, e\}$, be the sets of events that are recorded by measurement sites MS_1, MS_2 and MS_3, respectively. Assume that the set of observable events of local diagnoser LD_1 is $\Sigma_{o_1} = \{a, c\}$. Thus, $\Sigma^s_{o_1} = \{a_{s_1}, c_{s_1}\}$, where a_{s_1} and c_{s_1} denote the successful observation of events a and c, respectively, by local diagnoser LD_1. The occurrences of the events in Σ_{o_1} are transmitted through communication channels ch_{11} and ch_{12}, which implies that, $\Sigma_{o_{11}} = \{a\}$ and $\Sigma_{o_{12}} = \{c\}$. Assume now that the set of observable events of LD_2 is $\Sigma_{o_2} = \{b, c, e\}$. Thus, $\Sigma^s_{o_2} = \{b_{s_2}, c_{s_2}, e_{s_2}\}$, where b_{s_2}, c_{s_2}, and e_{s_2} denote the successful observation of events b, c, and e, respectively, by local diagnoser LD_2. The occurrences of the events in Σ_{o_2} are communicated through channels ch_{22} and ch_{23}, which implies that $\Sigma_{o_{22}} = \{c\}$ and $\Sigma_{o_{23}} = \{b, e\}$. Let σ_f be the failure event, and assume that the delay bounds of the communication channels are $k_{12} = 2$, $k_{23} = 1$ and $k_{11} = k_{22} = 0$.

Notice that automaton G generates failure traces $s_{F_1} = \sigma_f abec^p$ and $s_{F_2} = \sigma_f bcac^{p-1}$, and normal trace $s_N = bdac^p$, where $p \in \{1, 2, \ldots\}$. Since the sets of observable events of LD_1 and LD_2 are $\Sigma_{o_1} = \{a, c\}$ and $\Sigma_{o_2} = \{b, c, e\}$, respectively, and assuming that

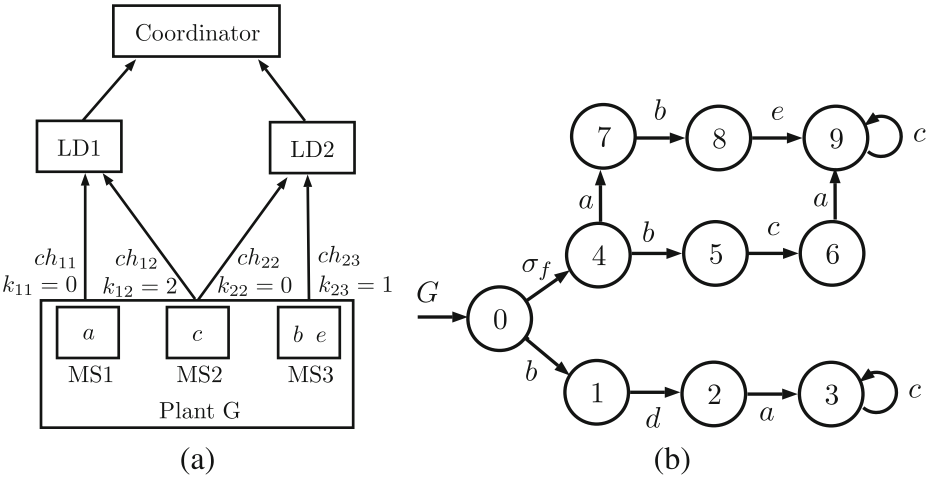


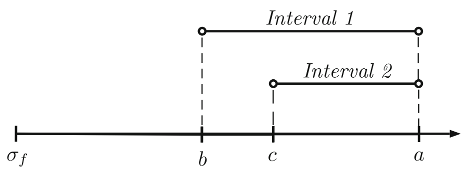
Fig. 4 Network diagnosis scheme (**a**) and automaton G (**b**) of Example 1

the system works perfectly, *i.e*, there is neither observation delays nor losses of events, the traces observed by LD_1 are $P_{o_1}(s_N) = P_{o_1}(s_{F_1}) = ac^p$ and $P_{o_1}(s_{F_2}) = cac^{p-1}$ and the traces observed by LD_2 are $P_{o_2}(s_N) = P_{o_2}(s_{F_2}) = bc^p$ and $P_{o_2}(s_{F_1}) = bec^p$. This implies that none of the local diagnosers can diagnose L_G alone. However, since LD_1 diagnosis trace s_{F_2}, and LD_2 diagnosis trace s_{F_1}, we conclude that the system is codiagnosable.

Let us now suppose that the plant generates trace $s = \sigma_f bca$ in the presence of communication delays. Since, $k_{23} = 1$ and $k_{22} = 0$, local diagnoser LD_2 may observe the occurrence of event b delayed by, at most, one step (Interval 1 of Fig. 5), and LD_2 observes the occurrence of event c without step delays, *i.e.*, after the occurrence of c and before the occurrence of event a (Interval 2 of Fig. 5). Notice that, since the occurrences of b and c are transmitted through different channels, and there is an intersection between Intervals 1 and 2, LD_2 may observe event c before observing event b. As a consequence, the following traces represent all possible observations of trace $s = \sigma_f bca$ by LD_2:

- Trace $b_{s_2}c_{s_2}$ that models the case when there is zero delay in the observation of event b, or the case when the delay in the observation of b is equal to one step, but LD_2 still receives the information about the occurrence of b before receiving the information about the occurrence of c;
- Trace $c_{s_2}b_{s_2}$ that models the case when the delay in the observation of b is equal to one step, and LD_2 receives the information about the occurrence of event c before receiving the information about the occurrence of b.

Fig. 5 Intervals 1 and 2, where LD_2 can observe events b and c, respectively, when trace $s = \sigma_f bca$ is generated by the system considered in Example 1



In order to obtain all possible observations of a trace $s \in L_G$ by a local diagnoser LD_i, we introduce a function that inserts events belonging to $\Sigma^s_{o_i}$ based on the communication delay bounds k_{ij} and event sets $\Sigma_{o_{ij}}$. Let us first define the following projections:

$$P_i : \Sigma_i^* \to \Sigma^*, \tag{5}$$

$$P_{i,o_{ij}} : \Sigma_i^* \to \Sigma_{o_{ij}}^*, \tag{6}$$

$$P_{i,s_{ij}} : \Sigma_i^* \to \Sigma_{o_{ij}}^{s*}. \tag{7}$$

In addition, let $w_{\sigma^{(l)}}$ denote the prefix of a trace $w \in \Sigma_i^*$ whose last event is the l-th occurrence of σ, and let $w_{\sigma_s^{(l)}}$ be the prefix of w whose last event is the l-th occurrence of σ_{s_i}, if $\sigma_{s_i}^{(l)} \in w$, or w, if $\sigma_{s_i}^{(l)} \notin w$.

Definition 3 (Insertion function) An insertion function associated with local diagnoser LD_i and observable events in $\Sigma_{o_{ij}}$, transmitted through communication channels ch_{ij} that have communication delay bound k_{ij}, $j = 1, 2, \ldots, m$, is a mapping

$$\begin{aligned} \chi_i : \Sigma^* &\to 2^{\Sigma_i^*} \\ s &\mapsto \chi_i(s) \end{aligned}$$

where $w \in \chi_i(s)$ if w satisfies the following conditions:

1. $P_i(w) = s$;
2. For all $\sigma \in \Sigma_{o_{ij}}$, and $\sigma^{(l)} \in w$:

$$\|P_i(w_{\sigma_s^{(l)}})\| - \|P_i(w_{\sigma^{(l)}})\| \leq k_{ij}, \tag{8}$$

3. For all $\sigma_{s_i} \in \Sigma^s_{o_{ij}}$, and $\sigma_{s_i}^{(l)} \in w$:

$$\sigma^{(l)} \in w_{\sigma_s^{(l)}}, \tag{9}$$

and

$$\|P_{i,o_{ij}}(w_{\sigma^{(l)}})\| = \|P_{i,s_{ij}}(w_{\sigma_s^{(l)}})\|, \tag{10}$$

The extension of χ_i to the domain 2^{Σ^*} is defined as $\chi_i(L) := \bigcup_{t \in L} \chi_i(t)$.

Condition 1 ensures that w is obtained from s by inserting events only from $\Sigma^s_{o_i}$. Condition 2 ensures that the delay between the occurrence of event $\sigma \in \Sigma_{o_{ij}}$, and its observation $\sigma_{s_i} \in \Sigma^s_{o_{ij}}$ is smaller than or equal to the maximum delay bound k_{ij}. Finally, condition 3 ensures that the observation of an event σ_{s_i} can only occur after event σ has occurred in s (Eq. 9), and that the observation of events transmitted through the same communication channel must be in the same order of their occurrence in s (Eq. 10). The following example illustrates the usefulness of insertion function χ_i.

Example 2 Consider the network decentralized diagnosis scheme depicted in Fig. 4a, and the plant modeled by automaton G depicted in Fig. 4b, where $\Sigma = \{a, b, c, d, e, \sigma_f\}$. Let us assume that $\Sigma_{o_2} = \{b, c, e\}$, which implies that $\Sigma_{o_{22}} = \{c\}$ and $\Sigma_{o_{23}} = \{b, e\}$, and that $k_{22} = 0$ and $k_{23} = 1$, *i.e.*, the observation of event c is not delayed, and the observation of events b and e can be delayed by at most one step for local diagnoser LD_2.

Let us assume that trace $s = \sigma_f bca \in L_G$ has been executed by the system. Then, by applying Definition 3, we obtain the following set of traces in Σ_2^* associated with all possible observations of trace s by local diagnoser LD_2 due to communication delays:

$$\chi_2(s) = \{\sigma_f bb_{s_2}cc_{s_2}a, \sigma_f bcb_{s_2}c_{s_2}a, \sigma_f bcc_{s_2}b_{s_2}a\}.$$

Notice that the projections in $\Sigma_{o_2}^s$ of the traces in $\chi_2(s)$ are, either $b_{s_2}c_{s_2}$ or $c_{s_2}b_{s_2}$, as expected.

Consider now trace $t = \sigma_f abec \in L_G$, and traces

$$w_1 = \sigma_f abee_{s_2}cb_{s_2}c_{s_2},\ w_2 = \sigma_f abeb_{s_2}e_{s_2}b_{s_2}cc_{s_2} \text{ and } w_3 = \sigma_f abee_{s_2}b_{s_2}cc_{s_2}.$$

Notice that traces w_1, w_2 and w_3 do not correspond to possible observations of trace t by local diagnoser LD_2 since: (i) the observation of event b, modeled by event b_{s_2}, is delayed by two steps in w_1, which is captured by condition 2 (Eq. 8); (ii) w_2 has two occurrences of event b_{s_2}, indicating that event b is observed twice by LD_2, which is not possible since there is only one occurrence of event b in trace t — this is captured by condition 3 (Eq. 9), and; (iii) the observation of events b and e are in an incorrect order in w_3, since these events are transmitted through the same communication channel and event e has occurred after event b in t, which is captured by condition 3 (Eq. 10).

In order to obtain an algorithm for the computation of automaton models G_i, $i = 1, \ldots, n$, such that $L(G_i) = \chi_i(L_G)$ we first propose an algorithm for the construction of automata D_i, $i = 1, \ldots, n$, that model the communication network between the plant and local diagnoser LD_i . In order to do so, the states of D_i must store the information about the occurrence of the events in Σ_{o_i} whose observations are being transmitted to local diagnoser LD_i, and the number of steps that have elapsed after the occurrence of these events. Thus, the states of D_i are labeled with traces formed with events in Σ_{o_i} and a symbol ν, that either represents the occurrence of an unobservable event in Σ_{uo_i}, or replaces an event in Σ_{o_i} that has been successfully observed by LD_i, *i.e.*, ν is used to represent a step delay when it is not important to memorize which event has occurred. Symbol ν is also used to denote the initial state of D_i, since, at this state, no event occurrence is being transmitted to LD_i.

In the following definition, we propose some operations over traces belonging to $(\Sigma_o \cup \{\nu\})^*$, and also define two functions that associate each event in Σ_o with its measurement site and its equivalent events in $\Sigma_{o_i}^s$, $i = 1, \ldots, n$.

Definition 4 Let $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$. Define $\Sigma_{o\nu} = \Sigma_o \cup \{\nu\}$ and the set of states Q, where each state $q \in Q$ is labeled with a trace $s \in \Sigma_{o\nu}^*$. Then, the following functions can be defined:

a. The replacement function rep is defined as:

$$rep : Q \times \mathbb{N} \to Q$$

where for all $q = q_1q_2...q_\ell \in Q$,

$$rep(q, i) = \begin{cases} q_1q_2...q_{i-1}\nu q_{i+1}...q_\ell, \text{ if } i \leq \ell \\ \text{undefined, otherwise.} \end{cases}$$

b. The elimination function cut is defined as:

$$cut : Q \to Q$$

where for all $q = q_1 q_2 ... q_\ell \in Q$,

$$cut(q) = \begin{cases} q_i q_{i+1} ... q_\ell, \text{ if } (\exists i \leq \ell)[(q_i \neq \nu) \wedge (q_k = \nu, \forall k \in \{1, 2, ..., i-1\})] \\ \nu, \text{ if } q_k = \nu, \forall k \in \{1, 2, \ldots, \ell\}. \end{cases}$$

c. The measurement site index function ms is defined as:

$$ms : \Sigma_{o\nu} \to \{1, 2, ..., m\}$$

where for all $\sigma \in \Sigma_{o\nu}$,

$$ms(\sigma) = \begin{cases} j : \text{ if } \sigma \in \Sigma_{o_{ij}} \text{ for some } i \in \{1, 2, ..., n\} \\ \text{undefined, otherwise.} \end{cases}$$

d. The bijective function ϕ_i is defined, for $i = 1, \ldots, n$, as:

$$\begin{aligned} \phi_i : \quad & \Sigma^s_{o_i} \to \Sigma_{o_i}, \\ & \sigma_{s_i} \mapsto \phi_i(\sigma_{s_i}) = \sigma, \end{aligned}$$

and ϕ_i is extended to sets of events as

$$\phi_i(\Sigma^s_{o_i}) = \bigcup_{\sigma_{s_i} \in \Sigma^s_{o_i}} \phi_i(\sigma_{s_i}).$$

According to Definition 4, function $rep(q, i)$ replaces the i-th element of state q with element ν. This function is introduced to represent that an event has occurred but the knowledge of which event has occurred is not important. Function $cut(q)$ eliminates the largest prefix of state q formed only with elements ν, and function $ms(\sigma)$ returns the index j which corresponds to the measurement site (MS_j) that detects the occurrence of event σ. Function $\phi_i(\sigma_{s_i})$ returns event σ whose successful observation is represented by σ_{s_i}.

Algorithm 1 describes the construction of automaton D_i, associated with local diagnoser LD_i, that models all possible delays in the communication of events to LD_i, from measurement site MS_j, $j = 1, 2, \ldots, m$. Automaton D_i will be referred to as the communication network delay model.

Notice that, Algorithm 1 can be divided in three parts: (i) initialization of automaton D_i, Steps 1 to 4.2.1, where we define the initial state and the associated transition functions; (ii) check of how many events can occur in the plant, with respect to communication delay k_{ij}, before one of them is observed, Step 4.2.7, and; (iii) modeling of observation of the events by LD_i, Step 4.2.9. The correctness of Algorithm 1 will be ensured by Lemma 2.

Algorithm 1 Construction of automaton D_i (Communication delay model)

Input: $m, n, \Sigma_{o_{ij}}, k_{ij}$, for $i = 1, \ldots, n$, $j = 1, \ldots, m$.
Output: $D_i = (Q_i, \Sigma_i, \delta_i, \Lambda_i, q_{0_i})$, $i = 1, \ldots, n$.

For $i = 1, 2, \ldots, n$

1: Define $q_{0_i} = \nu$ and $Q_i = \emptyset$.
2: Construct $\Sigma^s_{o_i}$ according to Eqs. (2) and (3), and define $\Sigma_i = \Sigma \cup \Sigma^s_{o_i}$.
3: $F \leftarrow (q_{0_i})$, where F denotes a FIFO queue.
4: While $F \neq \emptyset$ do

 4.1: $u \leftarrow head[F]$
 4.2: If $u = q_{0_i}$

  4.2.1: For all $\sigma \in \Sigma$,

   (a) Compute $\tilde{q} = \delta_i(u, \sigma) = \begin{cases} \sigma, & \text{if } \sigma \in \Sigma_{o_i}; \\ u, & \text{if } \sigma \in \Sigma_{uo_i}. \end{cases}$
   (b) If $\tilde{q} \neq u$, $Enqueue(F, \tilde{q})$

  4.2.2: $Q_i \leftarrow Q_i \cup \{u\}$
  4.2.3: $Dequeue(F)$

 Else

  4.2.4: Set $\ell = \|u\|$ and form set $I_\ell = \{1, 2, \ldots, \ell\}$
  4.2.5: Denote $u = \sigma_1\sigma_2\ldots\sigma_\ell$ and compute $I_\nu = \{y \in I_\ell : (\exists \sigma_y \in u)[\sigma_y = \nu]\}$
  4.2.6: Compute $I_{\ell\setminus\nu} = I_\ell \setminus I_\nu$
  4.2.7: For each $\sigma \in \Sigma_{o_i}$:

   (a) $\tilde{q} = \delta_i(u, \sigma) = \begin{cases} u\sigma, & \text{if } \|\sigma_y\sigma_{y+1}\ldots\sigma_\ell\| \leq k_{i,ms(\sigma_y)}, \forall y \in I_{\ell\setminus\nu} \\ \text{undefined, otherwise} \end{cases}$
   (b) If $\tilde{q}$ is defined, $Enqueue(F, \tilde{q})$

  4.2.8: For each $\sigma \in \Sigma_{uo_i}$:

   (a) $\tilde{q} = \delta_i(u, \sigma) = \begin{cases} u\nu, & \text{if } \|\sigma_y\sigma_{y+1}\ldots\sigma_\ell\| \leq k_{i,ms(\sigma_y)}, \forall y \in I_{\ell\setminus\nu} \\ \text{undefined, otherwise} \end{cases}$
   (b) If $\tilde{q} \notin F$, $Enqueue(F, \tilde{q})$

  4.2.9: For each $\Sigma^s_{o_{ij}}$, where $j = 1, 2, \ldots, m$:

   (a) Form set $Y = \{y : (\sigma_y \in u) \wedge (\sigma_y \in \phi_i(\Sigma^s_{o_{ij}})\}$
   (b) If $Y \neq \emptyset$, then compute $\hat{y} = min(Y)$ and $\tilde{q} = \delta_i(u, \phi_i^{-1}(\sigma_{\hat{y}})) = cut(rep(u, \hat{y}))$.
   (c) If $(\tilde{q} \notin Q_i) \wedge (\tilde{q} \notin F)$, $Enqueue(F, \tilde{q})$.

  4.2.10: Set $Q_i \leftarrow Q_i \cup \{u\}$
  4.2.11: $Dequeue(F)$

5: For each $q_i \in Q_i$, $\Lambda_i(q_i) = \{\sigma \in \Sigma_i : \delta_i(q_i, \sigma)!\}$

It is important to remark that Steps 4.2.9(a) and 4.2.9(b) are important when there are more than one event to be processed. Notice that, when more than one event of u belong to set $\Sigma_{o_{ij}}^{s}$, only the first occurred event is feasible to reach the new state defined in Step 4.2.9(b). This condition is a direct consequence of Assumption **A3**, which establishes that each communication channel follows FIFO rules, $i.e.$, there is no change in the order among events transmitted in the same communication channel.

Remark 2 In Qiu and Kumar (2008), a nondeterministic model is proposed to represent the effects of communication delays between local diagnosers in a distributed diagnosis architecture, assuming that there exists a unique delay bound k for all communication channels between diagnosers. This model was called k-$delaying$&$masking$ model. It is worth remarking that, differently from Qiu and Kumar (2008), we address here the problem of decentralized diagnosis using Protocol 3 of Debouk et al. (2000), assuming that each communication channel between a measurement site and a local diagnoser can have different delay bounds k_{ij}. The effects of these communication delays are captured by automaton D_i, computed according to Algorithm 1. It is also important to remark that, differently from the k-$delaying$&$masking$ model, the communication delay model D_i proposed here is deterministic.

The following example illustrates the construction of automaton D_i according to Algorithm 1.

Example 3 Consider the network decentralized diagnosis scheme depicted in Fig. 4a, and the plant modeled by automaton G depicted in Fig. 4b, where $\Sigma = \{a, b, c, d, e, \sigma_f\}$. Assume, as in Example 1, that the set of observable events of LD_1 and LD_2 are, respectively, $\Sigma_{o_1} = \{a, c\}$ and $\Sigma_{o_2} = \{b, c, e\}$. Thus, for local diagnoser LD_1, $\Sigma_{o_{11}} = \{a\}$ and $\Sigma_{o_{12}} = \{c\}$, and, for local diagnoser LD_2, $\Sigma_{o_{22}} = \{c\}$ and $\Sigma_{o_{23}} = \{b, e\}$. Assume that the system is subject to communication delays, where, as in Example 1, $k_{11} = k_{22} = 0$, $k_{12} = 2$ and $k_{23} = 1$. Then, for local diagnoser LD_1, the observation of occurrences of event c may be delayed by at most two steps, and, for local diagnoser LD_2, the observation of occurrences of events b and e can be delayed by at most one step.

In order to model the observation delays associated with LD_1 we need to construct automaton D_1, which is shown in Fig. 6a, by following the steps of Algorithm 1. In Step 1, the initial state q_{0_1} is defined as ν and the set of states Q_1 is defined as the empty set. In Step 2, sets $\Sigma_{o_1}^{s} = \{a_{s_1}, c_{s_1}\}$ and $\Sigma_1 = \{a, b, c, d, e, \sigma_f, a_{s_1}, c_{s_1}\}$ are formed. In Step 3, queue F is created and state $q_{0_1} = \nu$ is added to F. While queue F is not empty, the first element of F is assigned to variable u according to Step 4.1, and, since $u = \nu$, in Step 4.2.1, transitions from ν will be defined for all $\sigma \in \Sigma$, as follows: $\delta_1(\nu, a) = a$, $\delta_1(\nu, c) = c$ and $\delta_1(\nu, \sigma_f) = \delta_1(\nu, b) = \delta_1(\nu, d) = \delta_1(\nu, e) = \nu$. Next, states a and c are added to the end of queue F, that is $F = (\nu, a, c)$, and, in Step 4.2.2, state ν is added to set Q_1, $i.e.$, $Q_1 = \{\nu\}$. At Step 4.2.3, the first element of F is removed, and the queue becomes $F = (a, c)$.

In the second iteration, the first element of the queue is then assigned to variable u, $i.e$, $u = a$, and since u is different from ν in Step 4.2, the length of u is computed and assigned to variable ℓ and set $I_\ell = \{1\}$ is formed in Step 4.2.4. Then, sets $I_\nu = \emptyset$ and $I_{\ell \setminus \nu} = I_\ell$ are computed in Steps 4.2.5 and 4.2.6. Notice that, the conditions in Steps 4.2.7(a) and 4.2.8(a) check if the length of the suffixes of $u = \sigma_1 \sigma_2 \ldots \sigma_\ell$, is less than or equal to the delay of the communication channel that transmits event σ_y for all y in $I_{\ell \setminus \nu}$. Thus, in Steps 4.2.7(a) and 4.2.8(a), no transition from state a is defined, since channel ch_{11}, which transmits a, is not subject to communication delays. On the other hand, according to Step 4.2.9, a transition from state a to state ν labeled with a_{s_1} is created, which represents the successful observation of a by LD_1. To end this iteration, state a is added to set Q_1, *i.e.*, $Q_1 = \{\nu, a\}$ and is removed from queue F, which becomes $F = (c)$.

In the third iteration of Step 4, $u = c$. The length of u is computed and assigned to variable ℓ and set $I_\ell = \{1\}$ is formed. Then, sets $I_\nu = \emptyset$ and $I_{\ell \setminus \nu} = I_\ell$ are computed. Since channel ch_{12}, which transmits c, is subject to communication delays of at most two steps, in Step 4.2.7(a), transitions from state c are defined for all $\sigma \in \Sigma_{o_1}$ as follows: $\delta_1(c, a) = ca$ and $\delta_1(c, c) = cc$, and in Step 4.2.7(b), states ca and cc are added to the end of the queue F, *i.e.*, $F = (ca, cc)$. After this, in Step 4.2.8(a), transitions from state c are defined for all $\sigma \in \Sigma_{uo_1}$ as follows: $\delta_1(c, b) = \delta_1(c, d) = \delta_1(c, \sigma_f) = c\nu$, and in Step 4.2.8(b), state $c\nu$ is added to the end of the queue F, *i.e.*, $F = (ca, cc, c\nu)$. Step 4 will be repeated for all elements of queue F until $F = \emptyset$.

In order to model observation delays for LD_2, we need to construct automaton D_2 shown in Fig. 6b, which is constructed in a similar way as D_1. It is important to remark that, since

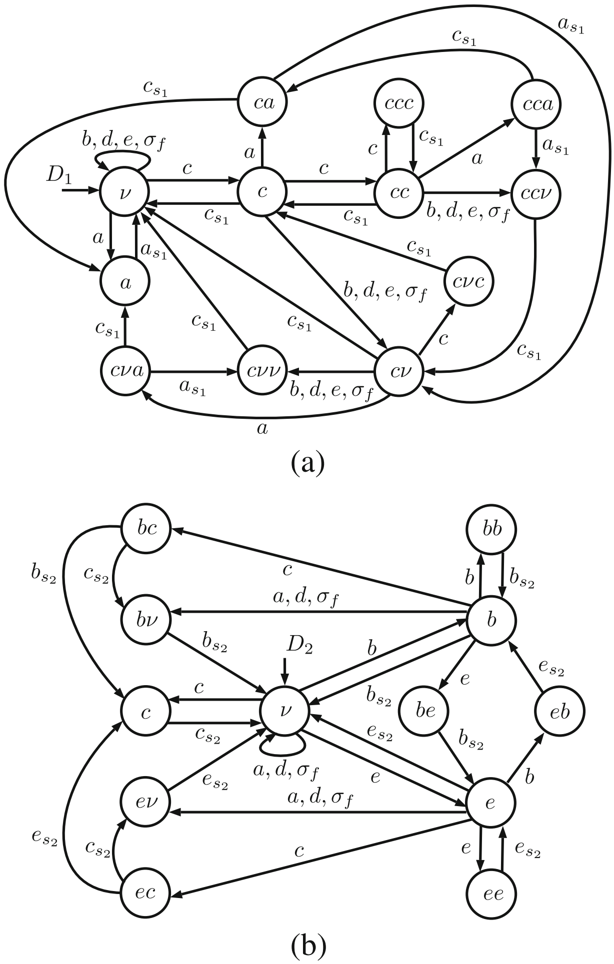


Fig. 6 Communication network delay model D_1 (**a**) and D_2 (**b**)

events b and e are communicated through the same channel ch_{23}, the order of observation of these events cannot be changed, *i.e.*, if substring be or eb is executed by the plant, then LD_2 receives $b_{s_2}e_{s_2}$ or $e_{s_2}b_{s_2}$, respectively, as shown in Fig. 6b.

Based on automaton D_i obtained in Algorithm 1, we may state the following results.

Lemma 1 *Let $w \in L(D_i)$, and define state $u = \delta_i(q_{0_i}, w)$. Then, $u = \nu$ if, and only if, every event $\sigma \in w$, where $\sigma \in \Sigma_{o_i}$, has its corresponding observation $\sigma_{s_i} \in w$. Otherwise, $u = \sigma_1\sigma_2 \ldots \sigma_l$, where $\sigma_1 \neq \nu$, and every event σ_y, $y = 1, 2, \ldots, l$, that is different from ν, accounts for the occurrence of event σ_y in w that belongs to Σ_{o_i} and has not been observed yet, with $l - y$ being equal to the number of events that have occurred in the system after the occurrence of σ_y.*

Proof The proof is done by induction in the length of the traces $w \in L(D_i)$.

Basis step. According to Step 1 of Algorithm 1, the initial state of D_i is equal to $q_{0_i} = \nu$. Thus, for $w = \epsilon$, $\delta_i(q_{0_i}, w) = \nu$, which agrees with the fact that there is no event in w, that belongs to Σ_{o_i}, whose observation has not been transmitted.

Induction hypothesis. For all $w \in L(D_i)$, such that $\|w\| \leq k$, $\delta_i(q_{0_i}, w) = \nu$ if, and only if, every event $\sigma \in w$, where $\sigma \in \Sigma_{o_i}$, has its corresponding observation $\sigma_{s_i} \in w$. Otherwise, $\delta_i(q_{0_i}, w) = \sigma_1\sigma_2 \ldots \sigma_l$, where $\sigma_1 \neq \nu$, and every event σ_y, $y = 1, 2, \ldots, l$, that is different from ν, accounts for the occurrence of event σ_y in w that belongs to Σ_{o_i} and has not been observed yet, with $l - y$ being equal to the number of events that have occurred in the system after the occurrence of σ_y.

Inductive step. Consider a trace $w\sigma \in L(D_i)$ such that $\|w\| = k$ and $\sigma \in \Sigma_i$.

Let us first consider the case that $\delta_i(q_{0_i}, w) = \nu$. Then, according to the induction hypothesis, w is such that every occurrence in w of events belonging to Σ_{o_i} are observed in w. Notice that, transitions from state ν of D_i are defined in Step 4.2.1 of Algorithm 1 for events $\sigma \in \Sigma$, and the reached state will be equal to σ, if $\sigma \in \Sigma_{o_i}$, or ν, if $\sigma \in \Sigma_{uo_i}$. Notice also that there are no transitions labeled by events in $\Sigma_{o_i}^s$ defined in state ν. Therefore, it can be concluded that the proposition of the lemma holds true for $\delta_i(q_{0_i}, w) = \nu$.

Let us now consider the case that $\delta_i(q_{0_i}, w) = \sigma_1\sigma_2 \ldots \sigma_l$. Thus, according to the induction hypothesis, every event σ_y, $y \in \{1, 2, \ldots, l\}$, that is different from ν, accounts for the occurrence of event σ_y in w that belongs to Σ_{o_i} and has not been observed yet, with $l - y$ being equal to the number of events that have occurred in the system after the occurrence of σ_y. According to Algorithm 1, the state reached from state $\delta_i(q_{0_i}, w)$ by a transition labeled by an event $\sigma \in \Sigma_i$ can be determined as follows:

(i) If $\sigma \in \Sigma_{o_i}$, then, according to Step 4.2.7, the reached state is

$$\delta_i(q_{0_i}, w\sigma) = \delta_i(q_{0_i}, w)\sigma = \sigma_1 \ldots \sigma_y \ldots \sigma_l\sigma.$$

(ii) If $\sigma \in \Sigma_{uo_i}$, then, according to Step 4.2.8, the reached state is

$$\delta_i(q_{0_i}, w\sigma) = \delta_i(q_{0_i}, w)\nu = \sigma_1 \ldots \sigma_y \ldots \sigma_l\nu.$$

(iii) If $\sigma \in \Sigma_{o_i}^s$, then, according to Step 4.2.9, the reached state is

$$\delta_i(q_{0_i}, w\sigma) = cut(rep(u, \hat{y})),$$

where, according to Step 4.2.9(a), $\hat{y}$ is the index of the first occurrence of event $\phi_i(\sigma)$ in $\delta_i(q_{0_i}, w)$ and, according to Definition 4, function rem replaces $\sigma_{\hat{y}}$ in $\delta_i(q_{0_i}, w)$ by ν, and function cut removes the largest prefix formed only with events ν.

Notice that, in cases *(i)* and *(ii)*, $\delta_i(q_{0_i}, w\sigma)$ is equal to the concatenation of $\delta_i(q_{0_i}, w)$ with σ and ν, respectively, which agrees with the fact that the number of event occurrences in the plant, after the occurrence of event σ_y, $y = 1, \ldots, l$, is increased by one after the occurrence of σ in the plant; in addition, in case *(i)*, $\delta_i(q_{0_i}, w)$ must be concatenated with σ, since $\sigma \in \Sigma_{o_i}$ and, clearly, its occurrence has not been observed in $w\sigma$, whereas, in case *(ii)*, $\delta_i(q_{0_i}, w)$ must be concatenated with ν since $\sigma \in \Sigma_{uo_i}$. In case *(iii)*, the occurrence of event $\sigma \in \Sigma_{o_i}^s$ represents the observation of an event in w, namely, it models the successful observation of event $\sigma_{\hat{y}}$ stored in $\delta_i(q_{0_i}, w)$. Thus, it is straightforward to conclude that we must replace $\sigma_{\hat{y}}$ by ν to obtain the reached state. This is done in Algorithm 1 by using function rem. In addition, function cut removes prefixes formed only by ν from $rep(u, \hat{y})$, *i.e.*, it guarantees that the first element of $\delta_i(q_{0_i}, w\sigma)$ is different from ν, if there is more than one element in $\delta_i(q_{0_i}, w)$ different from ν, or that $\delta_i(q_{0_i}, w\sigma) = \nu$, otherwise. In both cases, the lemma statement holds true, and the proof is complete. □

Lemma 2 $L(D_i) = \chi_i(\Sigma^*)$.

Proof Notice that, for a trace $w \in \Sigma_i^*$ to be in $\chi_i(s)$, where $s \in \Sigma^\star$, $P_i(w) = s$, and w must satisfy conditions 2 and 3 of Definition 3. Let us define language $A = \{w \in \Sigma_i^* :$ w satisfies conditions 2 and 3 of Definition 3$\}$. We will first prove that *(i)* $\chi_i(\Sigma^*) = A$ and, in the sequence, we will prove that *(ii)* $L(D_i) = A$.

(i) It is straightforward to conclude that $\chi_i(\Sigma^*) \subseteq A$ since, in accordance with Definition 3, every trace $w \in \chi_i(\Sigma^*)$ is such that $w \in \Sigma_i^*$ and satisfies conditions 2 and 3 of Definition 3, which implies that $w \in A$. On the other hand, since $A \subset \Sigma_i^*$, for all $w \in A$, there exists $s \in \Sigma^*$ such that $P_i(w) = s$. Therefore, w satisfies condition 1 of Definition 3 for $s = P_i(w)$, which implies that $w \in \chi_i(s)$, and, consequently, $w \in \chi_i(\Sigma^*)$.

(ii) Since, by the construction of D_i according to Algorithm 1, $L(D_i) \subseteq \Sigma_i^*$, we can show that $L(D_i) = A$ by proving that the following statement holds true:

$$\forall w \in \Sigma_i^*, \;\; w \in L(D_i) \Leftrightarrow w \text{ satisfies conditions 2 and 3 of Definition 3.} \tag{11}$$

The proof of Statement (11) is by induction in the length of strings $w \in \Sigma_i^*$.

Basis step. Let $w = \epsilon$. Then, w satisfies conditions 2 and 3 of Definition 3. Moreover, since the initial state of D_i is defined (which is equal to ν), we can conclude that $w \in L(D_i)$.

Induction hypothesis. For all $w \in \Sigma_i^*$ such that $\|w\| \leq k$, $w \in L(D_i)$ if, and only if, w satisfies conditions 2 and 3 of Definition 3.

Inductive step. Let $w\sigma \in \Sigma_i^*$ be such that $\|w\| = k$ and $\sigma \in \Sigma_i$.

Let us consider the case when $\delta_i(q_{0_i}, w) = \nu$. According to Lemma 1, if $\delta_i(q_{0_i}, w) = \nu$, then every event in w that belongs to Σ_{o_i} has been observed in w. Thus, if $\sigma \in \Sigma$, trace $w\sigma$ satisfies conditions 2 and 3 of Definition 3 since, in accordance with the induction hypothesis, w satisfies these conditions. On the other hand, if $\sigma \in \Sigma_{o_i}^s$, then trace $w\sigma$ does not satisfy Eq. 9 of condition 3. Notice that, according to Algorithm 1, a transition labeled by σ from state $\delta_i(q_{0_i}, w) = \nu$ is defined only if $\sigma \in \Sigma$ (Step 4.2.1). Therefore, when $\delta_i(q_{0_i}, w) = \nu$, trace $w\sigma \in L(D_i)$ if, and only if, $w\sigma$ satisfies conditions 2 and 3 of Definition 3.

Let us now consider that $\delta_i(q_{0_i}, w) = u = \sigma_1\sigma_2 \ldots \sigma_l$, and $\sigma \in \Sigma$. In this case, condition 3 of Definition 3 is satisfied for $w\sigma$, since it is satisfied for w. Thus, it remains to verify if condition 2 is also satisfied for trace $w\sigma$. In order to do so, let n_y denote the delay bound of the channel that communicates the occurrence of event σ_y to diagnoser LD_i, and consider the problem of evaluating the possibility of the occurrence of event $\sigma \in \Sigma$ before the observation of one of the events that form u. According to Steps 4.2.7(a) (if $\sigma \in \Sigma_{o_i}$) and 4.2.8(a) (if $\sigma \in \Sigma_{uo_i}$), this evaluation is made in a recursive way through the suffixes of u. Thus, the transition labeled with σ from state u is defined if, and only if, for all suffixes of u whose first element is not ν, the delay bound n_y of the first element σ_y is bigger than the length of the suffix. Notice that, in accordance with Lemma 1, verifying this condition is equivalent to check if every event in Σ_{o_i}, that has occurred in w and whose observation has not occurred, satisfies Eq. 8. In addition, since w satisfies condition 2, every event in w whose occurrence has been observed in w also satisfies Eq. 8. Therefore, we can conclude that, when $\sigma \in \Sigma$, $w\sigma \in L(D_i)$ if, and only if, $w\sigma$ satisfies conditions 2 and 3 of Definition 3.

Let us now consider the case when $\delta_i(q_{0_i}, w) = u = \sigma_1\sigma_2\ldots\sigma_l$, and $\sigma \in \Sigma_{o_i}^s$. In this case, $w\sigma$ also satisfies condition 2 of Definition 3, since w satisfies it. Thus, it remains to be verified if condition 3 holds true for trace $w\sigma$. In order to do so, consider the possibility of creating a transition from state u, labeled by event $\sigma \in \Sigma_{o_i}^s$, carried out in Step 4.2.9, which is repeated for each communication channel ch_{ij}, $j = 1, 2, \ldots, m$. In Step 4.2.9(a), the set of indexes Y is computed with respect to state u and set $\Sigma_{o_{ij}}^s$. Notice that, in accordance with Lemma 1, $w\sigma$ satisfies Eq. 9 if, and only if, there exists $\phi_i(\sigma)$ in u. Thus, when Y is nonempty, index $\hat{y} = min(Y)$ determines event $\sigma_{\hat{y}}$ that corresponds to the first event in u transmitted through communication channel ch_{ij}. Consequently, $\phi_i^{-1}(\sigma_{\hat{y}})$ is the unique event in $\Sigma_{o_{ij}}^s$ such that $w\phi_i^{-1}(\sigma_{\hat{y}})$ satisfies Eqs. 9 and 10, and, according to Step 4.2.9(b), it is also the unique event in $\Sigma_{o_{ij}}^s$ that is used to create a new transition from state u. Therefore, it can be concluded that, when $\sigma \in \Sigma_{o_i}^s$, $w\sigma \in L(D_i)$ if, and only if, $w\sigma$ satisfies conditions 2 and 3 of Definition 3. □

Based on Algorithm 1 and Lemmas 1 and 2, we can state the following result.

Theorem 1 *Let L denote a language defined over Σ and let χ_i be the insertion function defined with respect to communication delay bounds k_{ij} and event sets $\Sigma_{o_{ij}}$, for $j = 1, 2, \ldots, m$. Then, $\chi_i(L) = P_i^{-1}(L) \cap L(D_i)$.*

Proof ($\subseteq$) According to Lemma 2, $L(D_i) = \chi_i(\Sigma^*)$, which implies that $\chi_i(L) \subseteq L(D_i)$. In addition, from condition 1 of Definition 3, we can conclude that $\chi_i(L) \subseteq P_i^{-1}(L)$. Therefore, $\chi_i(L) \subseteq P_i^{-1}(L) \cap L(D_i)$.

($\supseteq$) Let $w \in P_i^{-1}(L) \cap L(D_i)$. Then, there exists $s \in L$ such that $s = P_i(w)$. Moreover, since $w \in L(D_i)$, according to Lemma 3, w satisfies conditions 2 and 3 of Definition 3, which implies that $w \in \chi_i(s) \subseteq \chi_i(L)$. Thus, it can be concluded that $\chi_i(L) \supseteq P_i^{-1}(L) \cap L(D_i)$.

□

After the computation of automata D_i, $i = 1, \ldots, n$, we can obtain automata G_i, $i = 1, 2, \ldots, n$, that model all possible ordering of observation of the traces of L_G by local diagnoser LD_i due to communication delays, by performing the parallel composition of automata G and D_i, *i.e.*:

$$G_i = G \| D_i = (X_i, \Sigma_i, f_i, \Gamma_i, x_{0_i}, \emptyset). \tag{12}$$

Notice that the observable event set of G_i is $\Sigma_{i_o} = \Sigma^s_{o_i}$ and not Σ_{o_i}, and its unobservable event set is $\Sigma_{i_{uo}} = \Sigma$, *i.e.*, the occurrence of an event $\sigma_{s_i} \in \Sigma^s_{o_i}$ represents the successful observation of event $\sigma \in \Sigma_{o_i}$ by the local diagnoser LD_i.

Since $G_i = G \| D_i$, then the language generated by G_i is given by:

$$L(G_i) = P_i^{-1}(L_G) \cap L(D_i), \tag{13}$$

where P_i is the projection defined in Eq. 5 and $L(D_i)$ denotes the language generated by automaton D_i.

The following results regarding the language generated by G_i can be stated.

Corollary 1 $L(G_i) = \chi_i(L_G)$.

Proof The proof comes directly from Theorem 1 and Eq. 13. □

Corollary 2 $L(G_i) \cap (\Sigma_i - \Sigma_f)^* = \chi_i(L_N)$.

Proof Notice that $L_N = L_G \cap (\Sigma \setminus \Sigma_f)^*$. Then, according to Theorem 1,

$$\begin{aligned} \chi_i(L_N) &= P_i^{-1}(L_N) \cap L(D_i) \\ &= P_i^{-1}(L_G \cap (\Sigma \setminus \Sigma_f)^*) \cap L(D_i) \\ &= P_i^{-1}(L_G) \cap P_i^{-1}((\Sigma \setminus \Sigma_f)^*) \cap L(D_i). \end{aligned}$$

Since, $P_i^{-1}((\Sigma \setminus \Sigma_f)^*) = (\Sigma_i \setminus \Sigma_f)^*$, and, according to Eq. 13, $L(G_i) = P_i^{-1}(L_G) \cap L(D_i)$, we can conclude that $\chi_i(L_N) = L(G_i) \cap (\Sigma_i \setminus \Sigma_f)^*$. □

A direct consequence of Corollary 2 is that all possible observations of the normal traces executed by the system, due to event communication delays, can be easily obtained from automaton G_i.

Let us now define the following projection

$$P_{is_i} : \Sigma_i^* \rightarrow \Sigma^{s*}_{o_i}. \tag{14}$$

The possible observations of a trace $s \in L_G$ by local diagnoser LD_i is represented in G_i as a set formed with those traces $t \in L(G_i)$ such that $P_i(t) = s$. Thus, $P_{is_i}(t)$ corresponds to a possible observation of s by LD_i, as illustrated in the following example.

Example 4 Consider the same plant and decentralized diagnosis architecture presented in Example 3. Automata G_1 and G_2, depicted in Fig. 7a and b, respectively, are computed according to Eq. 12 as $G_i = G \| D_i$, for $i = 1, 2$. The sets of observable events and unobservable events of G_1 are $\Sigma_{1_o} = \{a_{s_1}, c_{s_1}\}$, and $\Sigma_{1_{uo}} = \{a, b, c, d, e, \sigma_f\}$, respectively, and the sets of observable events and unobservable events of G_2 are $\Sigma_{2_o} = \{b_{s_2}, c_{s_2}, e_{s_2}\}$, and $\Sigma_{2_{uo}} = \{a, b, c, d, e, \sigma_f\}$, respectively.

Notice that, languages $L(G_1)$ and $L(G_2)$ represent all possible ordering of observation of traces $s \in L_G$ with respect to $\Sigma_{o_{ij}}$ and k_{ij}, for $j \in \{1, 2, 3\}$. For instance, let us consider the occurrence of trace $s = \sigma_f bca$ in the plant, and its possible observations by local diagnoser LD_2. The traces in $L(G_2)$ that are associated with the occurrence of s are those traces $t \in L(G_2)$ such that $P_2(t) = \sigma_f bca$, which are $t_1 = \sigma_f bb_{s_2}cc_{s_2}a$, $t_2 = \sigma_f bcb_{s_2}c_{s_2}a$, and $t_3 = \sigma_f bcc_{s_2}b_{s_2}a$. Thus, as expected, all possible observations of trace s by LD_2 are $b_{s_2}c_{s_2}$ and $c_{s_2}b_{s_2}$, since $P_{2s_2}(t_1) = P_{2s_2}(t_2) = b_{s_2}c_{s_2}$ and $P_{2s_2}(t_3) = c_{s_2}b_{s_2}$.

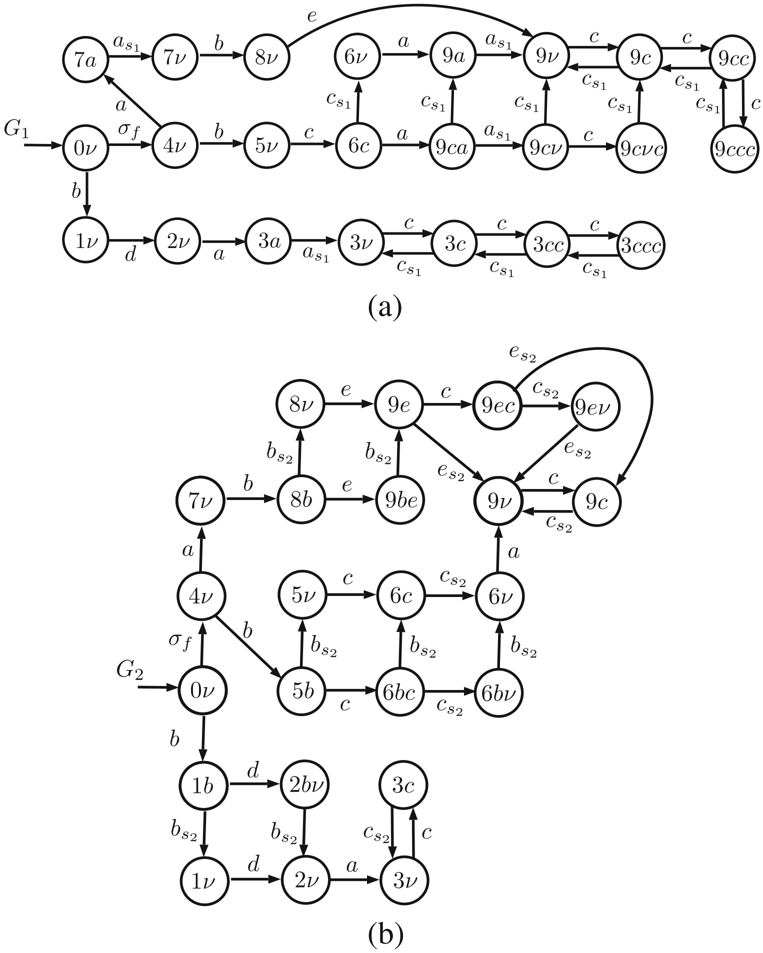


Fig. 7 Automaton G_1 (**a**) and automaton G_2 (**b**)

3.3 Model of the plant subject to communication delays and intermittent loss of observations

After the computation of automata G_i, for $i = 1, 2, \ldots, n$, that represent all possible observations by local diagnosers LD_i, $i = 1, 2, \ldots, n$, of the language generated by G due to communication delays of events, we will now model the intermittent loss of observation of events in the communication channels. In order to do so, we will use the dilation function introduced in Carvalho et al. (2012).

Consider the partition of the set of observable events associated with diagnoser LD_i, $\Sigma_{i_o} = \Sigma_{i,ilo} \dot{\cup} \Sigma_{i,nilo}$, where $\Sigma_{i,ilo}$ and $\Sigma_{i,nilo}$ denote, respectively, the set of events that are subject to intermittent loss of observation, and the set of events that are not subject to intermittent loss of observation. Let $\Sigma^s_{i,ilo} = \phi^{-1}(\Sigma_{i,ilo})$ and $\Sigma^s_{i,nilo} = \phi^{-1}(\Sigma_{i,nilo})$. Then, since the observable event set of G_i is given by $\Sigma_{i_o} = \Sigma^s_{o_i}$, we can make the following partition of the observable event set of G_i:

$$\Sigma^s_{o_i} = \Sigma^s_{i,ilo} \dot{\cup} \Sigma^s_{i,nilo}, \quad (15)$$

where the events of $\Sigma^s_{i,ilo}$ and $\Sigma^s_{i,nilo}$ denote the successful transmission to diagnoser L_{D_i} of the events of $\Sigma_{i,ilo}$ and $\Sigma_{i,nilo}$, respectively.

Let us define now the set of unobservable events that models the intermittent loss of observation of events $\sigma \in \Sigma^s_{i,ilo}$ as $\Sigma^{s'}_{i,ilo} = \{\sigma' : \sigma \in \Sigma^s_{i,ilo}\}$ and set $\Sigma'_i = \Sigma_i \cup \Sigma^{s'}_{i,ilo}$. Then, the dilation function $D_{s_i} : \Sigma^*_i \rightarrow 2^{(\Sigma'_i)^*}$ is defined in a recursive way as:

$$\begin{aligned} D_{s_i}(\epsilon) &= \{\epsilon\}, \\ D_{s_i}(\sigma) &= \begin{cases} \{\sigma\}, \text{ if } \sigma \in \Sigma_i \setminus \Sigma^s_{i,ilo} \\ \{\sigma, \sigma'\}, \text{ if } \sigma \in \Sigma^s_{i,ilo} \end{cases} \\ D_{s_i}(s_i\sigma) &= D_{s_i}(s_i)D_{s_i}(\sigma), \forall s_i \in \Sigma^*_i, \forall \sigma \in \Sigma_i. \end{aligned}$$

The dilation operation D_{s_i} is extended to languages in a straightforward way as $D_{s_i}(L) = \bigcup_{s \in L} D_{s_i}(s)$.

We can now obtain automaton G'_i that generates language $D_{s_i}[L(G_i)]$, and that models both, all possible ordering of observation of events $\sigma \in \Sigma_o$ due to communication delays and the intermittent loss of observation of events $\sigma \in \Sigma_{i,ilo}$. This automaton will be defined as follows:

$$G'_i = (X_i, \Sigma'_i, f'_i, \Gamma'_i, x_{0_i}, \emptyset),$$

where $\Gamma'_i(x_i) = D_{s_i}[\Gamma_i(x_i)]$, $\forall x_i \in X_i$, and $f'_i(x_i, \sigma') = f_i(x_i, \sigma)$, if $\sigma' \in \Sigma^{s'}_{i,ilo}$, and $f'_i(x_i, \sigma) = f_i(x_i, \sigma)$, if $\sigma \in \Sigma_i \setminus \Sigma^{s'}_{i,ilo}$. Notice that, if $\Sigma_{i,ilo} = \emptyset$, $G'_i = G_i$, which implies that $D_{s_i}[L(G_i)] = L(G_i)$.

The following example illustrates the construction of G'_i.

Example 5 Let us consider the problem addressed in Example 4, and assume that automata G_1 and G_2 have been calculated. In addition, suppose that event e is subject to intermittent loss of observation by local diagnoser LD_2. Thus, for local diagnoser LD_1, $\Sigma_{1,ilo} = \emptyset$ and $\Sigma_{1,nilo} = \{a, c\}$, which implies that automaton G'_1 is equal to automaton G_1 shown in Fig. 7a. For local diagnoser LD_2, $\Sigma_{2,ilo} = \{e\}$ and $\Sigma_{2,nilo} = \{b, c\}$. Automaton G'_2 that models the communication delay and intermittent loss of observations of the events in $\Sigma_{2,ilo}$ is shown in Fig. 8. Notice that, as expected, $L(G'_1) = D_{s_1}[L(G_1)] = L(G_1)$ and $L(G'_2) = D_{s_2}[L(G_2)]$.

3.4 Definition of network codiagnosability of discrete-event systems

The network codiagnosability of the language generated by a DES is defined as follows.

Definition 5 Let L_G and $L_N \subset L_G$ be the prefix-closed languages generated by G and G_N, respectively. Then, L_G is said to be network codiagnosable with respect to $\chi_i : \Sigma^* \rightarrow 2^{\Sigma^*_i}$, D_{s_i}, projection $P'_{s_i} : \Sigma'^*_i \rightarrow \Sigma^{s*}_{o_i}$, for $i = 1, \ldots, n$, and Σ_f if

$$\begin{aligned} &(\exists z \in \mathbb{N})(\forall s \in L_G \setminus L_N)(\forall st \in L_G \setminus L_N, ||t|| \geq z) \Rightarrow \\ &(\exists i \in \{1, \ldots, n\})[P'_{s_i}[D_{s_i}(\chi_i(st))] \cap P'_{s_i}[D_{s_i}(\chi_i(\omega_i))] = \emptyset, \forall \omega_i \in L_N]. \end{aligned}$$

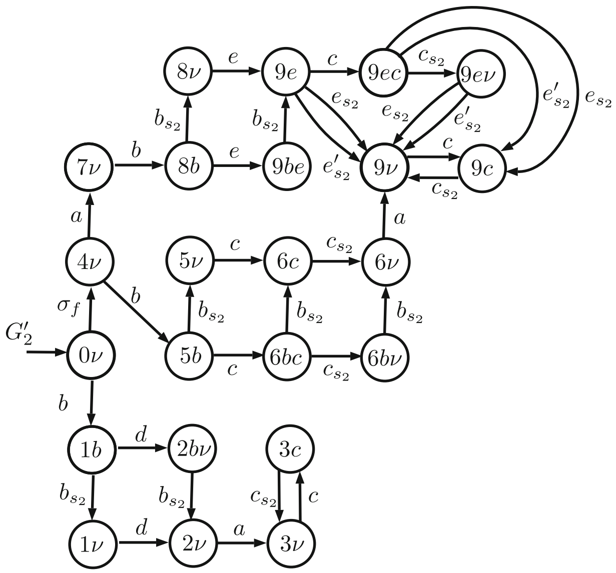


Fig. 8 Automaton G_2'

According to Definition 5, language L_G is not network codiagnosable if there exist a failure trace s and an arbitrarily long length trace t, such that there exist traces $s_i t_i \in D_{s_i}(\chi_i(st))$, $i = 1, 2, \ldots, n$, where $s_i t_i$ is not necessarily different from $s_j t_j$ for $i, j \in \{1, 2, \ldots, n\}$ and $s_{i_N} \in D_{s_i}(\chi_i(\omega_i))$, with $\omega_i \in L_N$, satisfying $P'_{s_i}(s_i t_i) = P'_{s_i}(s_{i_N})$, for all $i \in \{1, \ldots, n\}$. In words, a language L_G is not network codiagnosable if there exist a failure trace st, with arbitrarily long length after the occurrence of the failure event, and there exist normal traces ω_i, for $i = 1, \ldots, n$, such that, the change in the order of observation and the loss of observation of events create ambiguous observations in all local diagnosers.

4 Verification of network codiagnosability of discrete-event systems

We present in the sequel an algorithm for the verification of network codiagnosability of DES based on the verification algorithm proposed in Moreira et al. (2011). In order to do so, we first present the definition of the one-to-one event renaming function

$$\rho_i : \Sigma'_{i_N} \to \Sigma'_{i_\rho}, \qquad (16)$$
$$\sigma \mapsto \rho_i(\sigma) = \begin{cases} \sigma_{\rho_i}, & \text{if } \sigma \in (\Sigma \cup \Sigma^{s'}_{i,ilo}) \setminus \Sigma_f \\ \sigma, & \text{if } \sigma \in \Sigma^s_{o_i}. \end{cases}$$

where $\Sigma'_{i_N} = \Sigma'_i \setminus \Sigma_f$, for $i = 1, \ldots, n$. The domain of function ρ_i can be extended to $\Sigma'^*_{i_N}$ as usual, *i.e.*, $\rho_i(s\sigma) = \rho_i(s)\rho_i(\sigma)$, for all $s \in \Sigma'^*_{i_N}$ and $\sigma \in \Sigma'_{i_N}$. Function ρ_i can also be applied to a language K as $\rho_i(K) = \cup_{s \in K} \rho_i(s)$.

Algorithm 2 Network codiagnosability verification of DES

Input: $G'_i = (X_i, \Sigma'_i, f'_i, \Gamma'_i, x_{0_i}, \emptyset)$, for $i = 1, \ldots, n$.
Output: $G_V = (X_V, \Sigma_V, f_V, \Gamma_V, x_{0,V}, X_{V_m})$.

1: Compute automata $G'_{i,N} = (X'_{i_N}, \Sigma'_{i_N}, f'_{i_N}, \Gamma'_{i_N}, (x_{0_i}, N), \emptyset)$, where $\Sigma'_{i_N} = \Sigma'_i \setminus \Sigma_f$, for $i = 1, \ldots, n$, that model the normal behavior of G'_i as presented in Moreira et al. (2011).
2: Compute automata $G'_{i,F} = (X'_{i_F}, \Sigma'_i, f'_{i_F}, \Gamma'_{i_F}, (x_{0_i}, N), \emptyset)$, for $i = 1, \ldots, n$, that model the failure behavior of G'_i as presented in Moreira et al. (2011).
3: Construct automata $G'_{i,\rho} = (X'_{i_N}, \Sigma'_{i_\rho}, f'_{i_\rho}, \Gamma'_{i_\rho}, (x_{0_i}, N), \emptyset)$, for $i = 1, \ldots, n$, where $\Sigma'_{i_\rho} = \rho_i(\Sigma'_{i_N})$, and $f'_{i_\rho}(x_{i_N}, \sigma_{\rho_i}) = f'_{i_N}(x_{i_N}, \sigma)$ with $\sigma_{\rho_i} = \rho_i(\sigma)$, for all $\sigma \in \Sigma'_{i_N}$ and $x_{i_N} \in X'_{i_N}$.
4: Compute automata $\bar{V}_i = G'_{i,\rho} \| G'_{i,F} = (Y_{V_i}, \Sigma_{V_i}, f_{V_i}, \Gamma_{V_i}, y_{V_i,0}, \emptyset)$, for $i = 1, \ldots, n$, where $\Sigma_{V_i} = \Sigma'_{i_\rho} \cup \Sigma'_i$.
5: Find all cyclic paths $cl_i = (y^k_{V_i}, \sigma_k, y^{k+1}_{V_i}, \sigma_{k+1}, \ldots, \sigma_\ell, y^k_{V_i})$, where $\ell \geq k > 0$ in $\bar{V}_i$ that satisfy the following condition:

$$\begin{aligned} &\exists j \in \{k, k+1, \ldots, \ell\} \text{ such that, for some} \\ &y^j_{V_i} = (x^j_i, N, y^j_i, F) \wedge (\sigma_j \in \Sigma'_i) \end{aligned} \tag{17}$$

where $x^j_i, y^j_i \in X_i$.
6: Compute automata $V_i = (Y_{V_i}, \Sigma_{V_i}, f_{V_i}, \Gamma_{V_i}, y_{V_i,0}, Y_{V_i,m})$, where $Y_{V_i,m}$ is formed by the states of $\bar{V}_i$ that belong to the strongly connected components that contain cyclic paths cl_i satisfying condition (17).
7: Compute the verifier automaton $G_V = V_1 \| \ldots \| V_n = (X_V, \Sigma_V, f_V, \Gamma_V, x_{V,0}, X_{V_m})$, where $\Sigma_V = \bigcup_{i=1}^n \Sigma_{V_i}$.
8: Verify the existence of a cyclic path $cl = (x^k_V, \sigma_k, x^{k+1}_V, \sigma_{k+1}, \ldots, \sigma_\ell, x^k_V)$ in G_V, $\ell \geq k > 0$, that satisfies the following condition:

$$\begin{aligned} &x^q_V \in X_{V_m}, \forall q \in \{k, k+1, \ldots, \ell\}, \text{ and for some} \\ &q \in \{k, k+1, \ldots, \ell\}, \sigma_q \in \Sigma. \end{aligned} \tag{18}$$

If the answer is yes, then L_G is not network codiagnosable with respect to χ_i, D_{s_i}, P'_{s_i}, for $i = 1, \ldots, n$, and Σ_f. Otherwise, L_G is network codiagnosable.

Remark 3 Notice that the renamed events of verifier V_p are different from the renamed events of a verifier V_q, where $p \neq q$.

Lemma 3 *Let $G'_{i,N}$ and $G'_{i,F}$ be computed according to Steps 1 and 2 of Algorithm 2, respectively. Then, $L(G'_{i,F}) = \bigcup_{s \in \overline{L_G \setminus L_N}} D_{s_i}(\chi_i(s))$, and $L(G'_{i,N}) = \bigcup_{\omega \in L_N} D_{s_i}(\chi_i(\omega))$.*

Proof The proof is straightforward from the construction of G'_i, $G'_{i,N}$ and $G'_{i,F}$, and Theorem 1. □

Theorem 2 *Language L_G is network codiagnosable with respect to χ_i, D_{s_i}, P'_{s_i}, for $i = 1, \ldots, n$, and Σ_f if, and only if, there does not exist a cyclic path $cl = (x_V^k, \sigma_k, x_V^{k+1}, \sigma_{k+1}, \ldots, x_V^{\ell}, \sigma_\ell, x_V^k)$, $\ell \geq k > 0$ in G_V satisfying the following condition:*

$$x_V^q \in X_{V_m}, \forall q \in \{k, k+1, \ldots, \ell\}, \text{ and for some } q \in \{k, k+1, \ldots, \ell\}, \sigma_q \in \Sigma. \tag{19}$$

Proof ($\Rightarrow$) Suppose that language L_G is not network codiagnosable with respect to χ_i, D_{s_i}, P'_{s_i}, for $i = 1, \ldots, n$, and Σ_f. Thus, according to Definition 5, there exists at least one arbitrarily long length trace $st \in L_G \setminus L_N$ and traces $\omega_i \in L_N$, $i = 1, \ldots, n$, where ω_i is not necessarily distinct from ω_j, for $j = 1, \ldots, n$ and $i \neq j$, such that $P'_{s_i}[D_{s_i}(\chi_i(st))] \cap P'_{s_i}[D_{s_i}(\chi_i(\omega_i))] \neq \emptyset$ for all $i \in \{1, 2, \ldots, n\}$. Thus, according to Lemma 3, if L_G is not network codiagnosable, there exist traces $s_i t_i \in L(G'_{i,F})$ and $s_{i_N} \in L(G'_{i,N})$ such that, $P'_{s_i}(s_i t_i) = P'_{s_i}(s_{i_N})$ for all $i \in \{1, 2, \ldots, n\}$. As shown in Moreira et al. (2011), the existence of traces $s_i t_i$ and s_{i_N} such that $P'_{s_i}(s_i t_i) = P'_{s_i}(s_{i_N})$ for all $i \in \{1, 2, \ldots, n\}$, implies in the existence of a path p_i in V_i, that ends with a cyclic path cl_i that satisfies condition (17), whose associated trace $v_i \in L(V_i)$ satisfies $P_{V_i i}(v_i) = s_i t_i$ and $P_{V_i \rho}(v_i) = s_{i_{N\rho}}$, where $s_{i_{N\rho}} = \rho_i(s_{i_N})$, $P_{V_i i} : \Sigma_{V_i}^* \to \Sigma_i^*$ and $P_{V_i \rho} : \Sigma_{V_i}^* \to \Sigma_{i_\rho}^*$.

Notice that, if the states of the cyclic path cl_i are marked, then $v_i \in L_m(V_i)$, where $L_m(V_i)$ denotes the marked language of V_i. Since $G_V = \|_{i=1}^n V_i$, then $L_m(G_V) = \bigcap_{i=1}^n P_{V V_i}^{-1}[L_m(V_i)]$, where $P_{V V_i} : \Sigma_V^* \to \Sigma_{V_i}^*$. Thus, $\bigcap_{i=1}^n P_{V V_i}^{-1}(v_i) \subseteq L_m(G_V)$. Let $v \in \bigcap_{i=1}^n P_{V V_i}^{-1}(v_i)$. Since $v_i \in L_m(V_i)$, $P_{V_i i}(v_i) = s_i t_i$ and $P_i(s_i t_i) = st$, for all $i \in \{1, \ldots, n\}$, and the common events that synchronize the traces v_i, for $i = 1, \ldots, n$, in $\bigcap_{i=1}^n P_{V V_i}^{-1}(v_i)$ are in Σ, then there will be a cyclic path in G_V, associated with v with all states marked, with at least one transition labeled with an event $\sigma \in \Sigma$.

($\Leftarrow$) Suppose that there exists a path p in G_V that ends with a cyclic path cl that satisfies condition (19), and let $v \in L_m(G_V)$ be the trace associated with p. Notice that, since $G_V = \|_{i=1}^n V_i$, then $L_m(G_V) = \bigcap_{i=1}^n P_{V V_i}{}^{-1}[L_m(V_i)]$, and $P_{V V_i}(v) = v_i \in L_m(V_i)$, for $i = 1, 2, ..., n$. Notice also that, the common events of traces $v_i \in L_m(V_i)$, for $i = 1, 2, ..., n$, are events $\sigma \in \Sigma$. Thus, since condition (19) is verified, then at least one event in the cyclic path cl belongs to Σ, which implies that all traces v_i are associated with a path p_i that ends with a cyclic path cl_i, formed with marked states, that has an event in Σ. According to Algorithm 2, the states of a cyclic path cl_i in V_i are marked only if the failure has occurred. Thus, associated with the cyclic path cl of G_V there exists one cyclic path cl_i in each verifier V_i, for $i = 1, \ldots, n$, that satisfies condition (17), *i.e.*, there exists a failure trace $s_i t_i \in L(G_i)$, with arbitrarily long length, and a normal trace $s_{i_N} \in L(G_i)$, such that $P'_{s_i}(s_i t_i) = P'_{s_i}(s_{i_N})$, for all $i \in \{1, \ldots, n\}$. In order to show that L_G is not network codiagnosable, notice that, since condition (19) is verified, then there exists an arbitrarily long length failure trace $st \in \Sigma^*$, such that $P_V(v) = st$, where $P_V : \Sigma_V^* \to \Sigma^*$. Since the events in Σ are common events of all verifiers V_i and $G_V = \|_{i=1}^n V_i$, then $P_{V_i}(v_i) = st$, where $P_{V_i} : \Sigma_{V_i}^* \to \Sigma^*$, which shows that there exists an arbitrarily long length failure trace st such that $s_i t_i \in D_{s_i}(\chi_i(st))$ for $i \in \{1, \ldots, n\}$. Thus, according to Definition 5, L_G is not network codiagnosable with respect to χ_i, D_{s_i}, P'_{s_i}, for $i = 1, \ldots, n$, and Σ_f. □

Example 6 Let us verify the network codiagnosability of the system presented in Example 1. Following Steps 1 to 3 of Algorithm 2, automata $G'_{1,\rho}$ and $G'_{1,F}$, shown in Fig. 9a and b, respectively, and automata $G'_{2,\rho}$ and $G'_{2,F}$, shown in Fig. 10a and b, respectively, are computed. In Steps 4 to 6 of Algorithm 2, verifiers V_1 and V_2 are computed. Due to the size of these automata, we show in Fig. 11a and b only one path of V_1 and V_2, respectively, that contain cyclic paths, referred to as cl_1 and cl_2, that satisfy condition (17). After the computation of V_1 and V_2, automaton $G_V = V_1||V_2$ can be computed, in accordance with Step 7. We show in Fig. 12 only the path of G_V that contains a cyclic path cl associated with the cyclic paths cl_1 and cl_2. Notice that cl is formed by marked states and contains an event $c \in \Sigma$. Thus, according to condition (18), language L_G is not network codiagnosable with respect to $\chi_i : \Sigma^* \to 2^{\Sigma_i^*}$, D_{s_i}, $P'_{s_i} : \Sigma_i'^* \to \Sigma_{o_i}^{s*}$, for $i = 1, 2$, and Σ_f.

5 Complexity analysis of algorithm 2

The computational complexity in the construction of verifier G_V, according to Algorithm 2, depends on the complexity of the computation of automata D_i, G_i, G'_i, and V_i, for $i = 1, \ldots, n$. Table 1 shows the maximum number of states and transitions of the automata computed in order to obtain the verifier automaton G_V for n local diagnosers according to Algorithm 2.

In the first step for the construction of automaton D_i according to Algorithm 1, only one initial state is created. Then, from the initial state, $|\Sigma_o|$ states can be reached in the worst case, and for each one of these states, $|\Sigma_o| + 1$ states can be reached. The number of states created at each step of the construction of D_i depends on the delays of the communication channels. Thus, assuming that the maximum delay for all communication channels is k, then, in the worst case, the number of states of automaton D_i is

$$|X_{D_i}| = 1 + \left[\sum_{j=0}^{k} (|\Sigma_o| + 1)^j\right] \times |\Sigma_o|. \tag{20}$$

Since D_i is a deterministic automaton, then the maximum number of transitions of D_i is equal to $|X_{D_i}| \times (|\Sigma| + |\Sigma_o|)$.

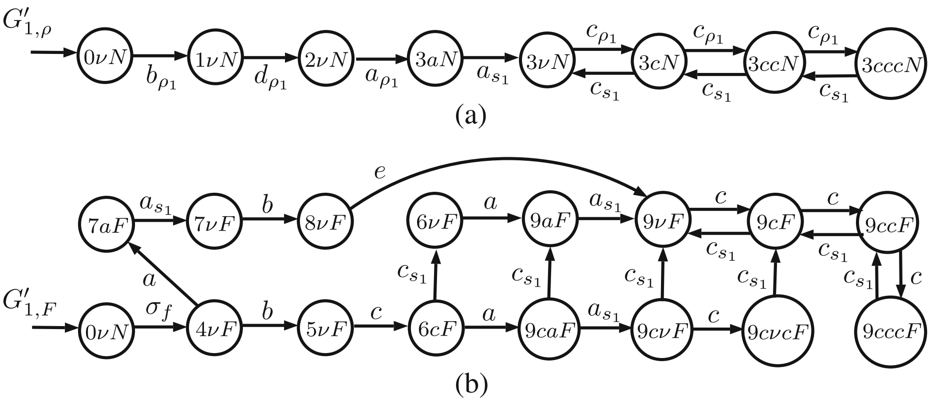


Fig. 9 Automata $G'_{1,\rho}$ (**a**) and $G'_{1,F}$ (**b**)

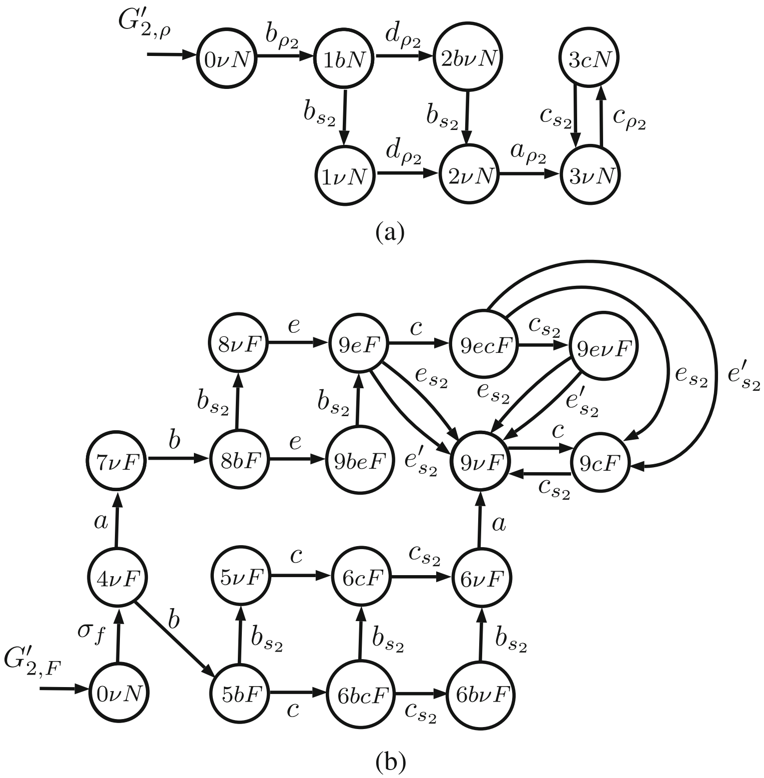


Fig. 10 Automata $G'_{2,\rho}$ (**a**) and $G'_{2,F}$ (**b**)

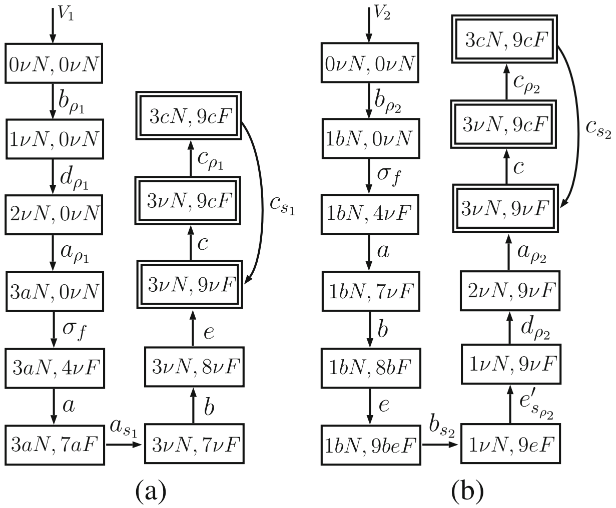


Fig. 11 Path of V_1 with cyclic path cl_1 embedded (**a**), and path of V_2 with cyclic path cl_2 embedded (**b**)

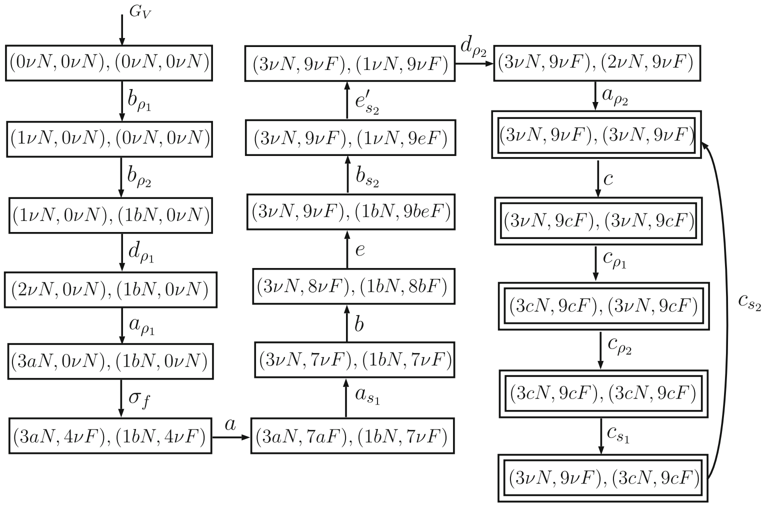


Fig. 12 Path of G_V with an embedded cyclic path cl that violates the network codiagnosability of L_G

Automaton G_i is computed by the parallel composition of automata G and D_i. Since $|X|$ is the number of states of G, then the number of states and transitions of G_i are, respectively, $|X| \times |X_{D_i}|$ and $|X| \times |X_{D_i}| \times |\Sigma_i|$.

Since automaton G'_i is computed by introducing a transition labeled with an event $\sigma' \in \Sigma^{s\prime}_{i,ilo}$ in parallel with the transitions of G_i labeled with $\sigma \in \Sigma^{s}_{i,ilo}$, the number of states and transitions of G'_i are, in the worst case, $|X| \times |X_{D_i}|$ and $|X| \times |X_{D_i}| \times |\Sigma'_i|$, respectively. Since $\Sigma'_i = \Sigma \cup \Sigma^s_{o_i} \cup \Sigma^{s\prime}_{i,ilo}$, the maximum number of events in Σ'_i is $|\Sigma| + 2 \times |\Sigma_o|$. Thus, the number of transitions in G'_i is, in the worst case, $|X| \times |X_{D_i}| \times (|\Sigma| + 2 \times |\Sigma_o|)$.

Table 1 Computational Complexity of Algorithm 2

	Number of states	Number of transitions
G	$\lvert X\rvert$	$\lvert X\rvert \times \lvert\Sigma\rvert$
D_i	$\lvert X_{D_i}\rvert = 1 + \left[\sum_{j=0}^{k} (\lvert\Sigma_o\rvert + 1)^j\right] \times \lvert\Sigma_o\rvert$	$\lvert X_{D_i}\rvert \times (\lvert\Sigma\rvert + \lvert\Sigma_o\rvert)$
G_i	$\lvert X\rvert \times \lvert X_{D_i}\rvert$	$\lvert X\rvert \times \lvert X_{D_i}\rvert \times (\lvert\Sigma\rvert + \lvert\Sigma_o\rvert)$
G'_i	$\lvert X\rvert \times \lvert X_{D_i}\rvert$	$\lvert X\rvert \times \lvert X_{D_i}\rvert \times (\lvert\Sigma\rvert + 2\lvert\Sigma_o\rvert)$
V_i	$2\left(\lvert X\rvert \times \lvert X_{D_i}\rvert\right)^2$	$2\left(\lvert X\rvert \times \lvert X_{D_i}\rvert\right)^2 \times (2\lvert\Sigma\rvert + 3\lvert\Sigma_o\rvert - \lvert\Sigma_f\rvert)$
G_v	$2^n \times \lvert X\rvert^{2n} \times \prod_{i=1}^{n} \lvert X_{D_i}\rvert^2$	$2^n \times \lvert X\rvert^{2n} \times \prod_{i=1}^{n} \lvert X_{D_i}\rvert^2 \times \left[(n+1)\lvert\Sigma\rvert + 3n\lvert\Sigma_o\rvert - n\lvert\Sigma_f\rvert\right]$
Complexity		$O\left(n \times 2^{4n} \times \lvert X\rvert^{2n} \times \lvert\Sigma\rvert^{2n+1} \times (\lvert\Sigma\rvert + 1)^{2nk}\right)$

Following the complexity analysis presented in Moreira et al. (2011), each verifier V_i has, in the worst case, $2 \times (|X| \times |X_{D_i}|)^2$ states and $2 \times (|X| \times |X_{D_i}|)^2 \times (2 \times |\Sigma| + 3 \times |\Sigma_o| - |\Sigma_f|)$ transitions. Thus, since $G_V = \|_{i=1}^n V_i$, the maximum number of states and transitions of G_V are, respectively, $2^n \times |X|^{2n} \times \Pi_{i=1}^n |X_{D_i}|^2$ and $2^n \times |X|^{2n} \times \Pi_{i=1}^n |X_{D_i}|^2 \times [(n+1)|\Sigma| + 3n|\Sigma_o| - n|\Sigma_f|]$. We can conclude, from Eq. 20, that the complexity of Algorithm 2 is $O(n \times 2^{4n} \times |X|^{2n} \times |\Sigma|^{2n+1} \times (|\Sigma|+1)^{2nk})$, *i.e.*, it grows exponentially with the number of local diagnosers n and maximum communication delay k. From the authors knowledge there is no other way of obtaining a network delay model guaranteeing the same features and with the same modeling power as the one presented in this work. It is also important to remark that the intermittent loss of observations does not significantly increase the computational complexity, since the dilation operation, in the worst case, multiplies by two the number of observable transitions of automata G_i, $i = 1, \ldots, n$.

6 Conclusions

In this work, we address the problem of language codiganosability of networked DES subject to event communication delays and loss of observation. A necessary and sufficient condition for the network codiagnosability of the language generated by the system with respect to communication delays and loss of observation is presented. In addition, we propose an algorithm to verify this property. The computational complexity of the proposed algorithm is also presented.

References

Alves MVS, Basilio JC, Cunha AEC, Carvalho LK, Moreira MV (2014) Robust supervisory control against intermittent loss of observations. In: 12th workshop on discrete event systems, vol 12. Elsevier, Cachan, pp 294–299

Athanasopoulou E, Lingxi L, Hadjicostis C (2010) Maximum likelihood failure diagnosis in finite state machines under unreliable observations. IEEE Trans Autom Control 55(3):579–593

Balemi S (1994) Input/output discrete event processes and communication delays. Discrete Event Dyn Syst 4(1):41–85

Carvalho LK, Basilio JC, Moreira MV (2012) Robust diagnosis of discrete event systems against intermittent loss of observations. Automatica 48(9):2068–2078

Carvalho LK, Moreira MV, Basilio JC (2011) Generalized robust diagnosability of discrete event systems. In: 18th IFAC World Congress, pp 8737–8742, Milan. Elsevier

Carvalho LK, Moreira MV, Basilio JC, Lafortune S (2013) Robust diagnosis of discrete-event systems against permanent loss of observations. Automatica 49(1):223–231

Cassandras CG, Lafortune S (2008) Introduction to discrete event systems, 2nd edn. Springer, New York

Debouk R, Lafortune S, Teneketzis D (2000) Coordinated decentralized protocols for failure diagnosis of discrete event systems. Discrete Event Dyn Syst 10(1):33–86

Debouk R, Lafortune S, Teneketzis D (2003) On the effect of communication delays in failure diagnosis of decentralized discrete event systems. Discrete Event Dyn Syst 13(3):263–289

Huo Z, Fang H, Ma C (2004) Networked control system: state of the art. In: Proceedings of the 5th world congress on intelligent control and automation. IEEE, Hangzhou, pp 1319–1322

Lin F (2014) Control of networked discrete event systems: dealing with communication delays and losses. SIAM J Control Optim 52(2):1276–1298

Moreira MV, Basilio JC, Cabral FG (2016) Polynomial time verification of decentralized diagnosability of discrete event systems versus decentralized failure diagnosis of discrete event systems: a critical appraisal. IEEE Trans Autom Control 61(1):178–181

Moreira MV, Jesus TC, Basilio JC (2011) Polynomial time verification of decentralized diagnosability of discrete event systems. IEEE Trans Autom Control 56(7):1679–1684

Nunes C, Moreira MV, Alves MVS, Basilio JC (2016) Network codiagnosability of discrete-event systems subject to event communication delay. In: 13th workshop on discrete event systems. IEEE Xian, pp 217–223

Park SJ, Cho KH (2006) Delay-robust supervisory control of discrete event systems with bounded communication delays. IEEE Trans Autom Control 51(5):911–915

Park S-J, Cho K-H (2007a) Decentralized supervisory control of discrete event systems with communication delays based on conjunctive and permissive decision structures. Automatica 43(4):738–743

Park S-J, Cho K-H (2007b) Supervisory control of discrete event systems with communication delays and partial observations. Syst Control Lett 56(2):106–112

Qiu W, Kumar R (2006) Decentralized failure diagnosis of discrete event systems. IEEE Trans Syst Man Cybern Part A 36(2):384–395

Qiu W, Kumar R (2008) Distributed diagnosis under bounded delay communication of immediately forwarded local observations. IEEE Trans Syst Man Cybern Part A 38(3):628–642

Rohloff KR (2005) Sensor failure tolerant supervisory control. In: 44th IEEE conference on decision and control. IEEE, Seville, pp 3493–3498

Sadid WH, Ricker L, Hashtrudi-Zad S (2015) Robustness of synchronous communication protocols with delay for decentralized discrete-event control. Discrete Event Dyn Syst 25(1):159–176

Sampath M, Sengupta R, Lafortune S, Sinnamohideen K, Teneketzis D (1995) Diagnosability of discrete-event systems. IEEE Trans Autom Control 40(9):1555–1575

Sánchez AM, Montoya F (2006) Safe supervisory control under observability failure. Discrete Event Dyn Syst 16(4):493–525

Shu S, Lin F (2014) Decentralized control of networked discrete event systems with communication delays. Automatica 50(8):2108–2112

Shu S, Lin F (2015) Supervisor synthesis for networked discrete event systems with communication delays. IEEE Trans Autom Control 60(8):2183–2188

Takai S (2012) Verification of robust diagnosability for partially observed discrete event systems. Automatica 48(8):1913–1919

Tripakis S (2004) Decentralized control of discrete-event systems with bounded or unbounded delay communication. IEEE Trans Autom Control 49(9):1489–1501

Ushio T, Takai S (2016) Nonblocking supervisory control of discrete event systems modeled by mealy automata with nondeterministic output functions. IEEE Trans Autom Control 61(3):799–804

Carlos E. V. Nunes was born on July 4th, 1984 in Aracaju, Sergipe, Brazil. He received the Electronic Engineer and the M.Sc. degrees in Control at the Federal University of Sergipe in 2009 and 2012, respectively, and the D.Sc. degree from the Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, in 2016. He is currently an Associate Professor at the Department of Electrical Engineering of the Federal University of Bahia, Salvador. His main interests are supervisory control and failure diagnosis of discrete event systems.

Marcos V. Moreira was born on May 11, 1976 in Rio de Janeiro, Brazil. He received the Electrical Engineer degree, the M.Sc. degree and the D. Sc. degree in Control from the Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, in 2000, 2002 and 2006, respectively. Since 2007, he has been an Associate Professor at the Department of Electrical Engineering at the Federal University of Rio de Janeiro. His main interests are robust failure diagnosis of discrete-event systems, cyber-attacks, smart grids, and the development of control laboratory techniques.

Marcos V. S. Alves was born on March 5th, 1988 in Aracaju, Sergipe, Brazil. He received the Electronic Engineer degree at the Federal University of Sergipe in 2011 and the M.Sc. and D.Sc. degrees in Control from the Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, in 2014 and 2017 respectively. His main interests are supervisory control, failure diagnosis of discrete event systems, and cyber-physical system security.

Lilian K. Carvalho was born on March 11, 1979 in São Paulo, Brazil. She received the Electronic Engineer degree, the M.Sc. degree and the D. Sc. degree in Control from the Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, in 2003, 2005 and 2011, respectively. Since 2011, she has been an Associate Professor at the Department of Electrical Engineering at the Federal University of Rio de Janeiro. From September, 2014, to December, 2015, she spent a sabbatical year at the University of Michigan, Ann Arbor. Her main interests are fault diagnosis of discrete-event systems, cyber-attacks and the development of control laboratory techniques.

João Carlos Basilio was born on March 15, 1962 in Juiz de Fora, Brazil. He received the Electrical Engineering degree in 1986 from the Federal University of Juiz de Fora, Juiz de Fora, Brazil, the M.Sc. degree in Control from the Military Institute of Engineering, Rio de Janeiro, Brazil, in 1989, and the Ph.D. degree in Control from Oxford University, Oxford, U.K., in 1995. He began his career in 1990 as an Assistant Lecturer at the Department of Electrical Engineering of the Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, and, since 2014, has been a Full Professor in Control at the same department. He served as Academic Chair for the Control and Automation Engineering course of Polytechnic School of the Federal University of Rio de Janeiro from January, 2005, to December, 2006, as Chair for the Electrical Engineering Postgraduation Program (COPPE) from January, 2008, to February, 2009, as Head of the Electrical Engineering Department, from May, 2012 to February, 2014, and since 2014 he has been the Dean of Polytechnic School. From September, 2007, to December, 2008, he spent a sabbatical leave at the University of Michigan, Ann Arbor, and was an Invited Professor of École Centrale of Lille, University of Lille, France, during September, 2016. His current interests are fault diagnosis and supervisory control of discrete-event systems. Prof. Basilio is the recipient of the Correia Lima Medal.