An Event-based Distributed Diagnosis Framework using Structural Model Decomposition

Anibal Bregon[a,1,*], Matthew Daigle[b,2], Indranil Roychoudhury[c,2], Gautam Biswas[d], Xenofon Koutsoukos[d], Belarmino Pulido[a,1]

[a]*Department of Computer Science, University of Valladolid, Valladolid, 47011, Spain*
[b]*NASA Ames Research Center, Moffett Field, CA, 94035, USA*
[c]*SGT Inc., NASA Ames Research Center, Moffett Field, CA, 94035, USA*
[d]*Institute for Software Integrated Systems, Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN, 37235, USA*

Abstract

Complex engineering systems require efficient on-line fault diagnosis methodologies to improve safety and reduce maintenance costs. Traditionally, diagnosis approaches are centralized, but these solutions do not scale well. Also, centralized diagnosis solutions are difficult to implement on increasingly prevalent distributed, networked embedded systems. This paper presents a distributed diagnosis framework for physical systems with continuous behavior. Using Possible Conflicts, a structural model decomposition method from the Artificial Intelligence model-based diagnosis (DX) community, we develop a distributed diagnoser design algorithm to build local event-based diagnosers. These diagnosers are constructed based on global diagnosability analysis of the system, enabling them to generate local diagnosis results that are globally correct without the use of a centralized coordinator. We also use Possible Conflicts to design local parameter estimators that are integrated with the local diagnosers to form a comprehensive distributed diagnosis framework. Hence, this is a fully distributed approach to fault detection, isolation, and identification. We evaluate the developed scheme on a four-wheeled rover for different design scenarios to show the advantages of using Possible Conflicts, and generate on-line diagnosis results in simulation to demonstrate the approach.

Keywords: Distributed Diagnosis, Structural Model Decomposition, Discrete Event Systems, Possible Conflicts

1. Introduction

The need for increased performance, safety, and reliability of complex engineering systems motivates the development of efficient fault diagnosis methodologies. Fault diagnosis is fundamental to reduce downtime and increase system availability through the life of the system. The process of fault diagnosis includes timely fault detection, i.e., recognizing that a fault has occurred in the system; quick fault isolation, i.e., determining the root causes of the fault; and accurate fault identification, i.e., estimation of the fault size. Our focus in this work is on model-based approaches to on-line fault detection, isolation, and identification (FDII) in complex dynamic systems. An advantage of using model-based techniques against other diagnosis approaches, like expert systems or machine learning, lies in the reusability of models and the diagnostic algorithms [1]. In particular, in this work we focus on the consistency approach to model-based diagnosis (known as consistency-based diagnosis, CBD), which has seen significant research activities from the Artificial Intelligence diagnosis (DX) [2, 3] community in the last two decades.

[*] Corresponding author

Email addresses: `anibal@infor.uva.es` (Anibal Bregon), `matthew.j.daigle@nasa.gov` (Matthew Daigle), `indranil.roychoudhury@nasa.gov` (Indranil Roychoudhury), `gautam.biswas@vanderbilt.edu` (Gautam Biswas), `xenofon.koutsoukos@vanderbilt.edu` (Xenofon Koutsoukos), `belar@infor.uva.es` (Belarmino Pulido)

[1] A. Bregon and B. Pulido's work has been partially supported by the Spanish MCI TIN2009-11326 grant.

[2] M. Daigle and I. Roychoudhury's work has been partially supported by the NASA System-wide Safety and Assurance Technologies (SSAT) project.

Typically, centralized diagnosis solutions have been proposed for model-based diagnosis, but these solutions have several inherent shortcomings. First, if the centralized diagnoser fails, the system will have to operate without a diagnosis system (this is usually known as a single point of failure), and second, centralized solutions do not scale well as the size of the system increases [4–6]. Further, the increased ubiquity of distributed, networked systems makes the use of centralized diagnosis solutions unwieldy. These shortcomings encourage the development of distributed diagnosis frameworks for complex dynamic systems.

Our approach has its roots in a CBD qualitative fault isolation (QFI) framework [7], where fault detection involves statistical testing of discrepancies (known as residuals) between observed and expected system behavior, and fault isolation is performed by analyzing the qualitative values of these residuals. In previous work, we developed a distributed diagnosis approach [8], where a set of local distributed diagnosers are designed to provide globally correct local diagnosis results, without a centralized coordinator, and with minimal communication among the local diagnosers. Subsequently, this work was integrated into the formal event-based framework developed in [9] which improved diagnosability and efficiency of the local diagnosers [10]. However, the approach proposed in [10] only distributes the fault isolation task, but the fault detection and identification tasks are still centralized because they use a global model of the system.

Model decomposition methods provide a systematic approach to decompose the diagnosis task. Several approaches have been proposed to decompose the global system model into submodels that contain sufficient analytical redundancy to perform fault detection [11–13]. In this work, we focus on the *Possible Conflicts* (PCs) [14] approach. The PCs approach is a structural model decomposition technique from the DX community. PCs are computed as subsets of equations containing the minimal number of constraints required to estimate a measurement. PCs decompose the system model into independent submodels by using measured signals of the global model as local inputs for the submodels. Therefore, PCs provide the subset of constraints from the global system model required to compute residuals, and each PC can operate independently, providing a natural way to distribute the residual generation process. Moreover, the PCs also provide the mechanisms to decompose the global fault identification task into reduced size local parameter estimators, that can be distributed [15]. However, PCs still require the use of a centralized coordinator for fault isolation to compute the set of diagnoses based on the triggered PCs.

In this work, we start from a common framework for diagnosis (CBD with QFI), and introducing structural model decomposition with PCs, we develop an on-line distributed diagnosis framework to design local diagnosers that fully distribute the diagnosis process. The primary contributions of this work are as follows:

1. A unified distributed diagnosis framework that covers fault detection, isolation, and identification, and is able to perform on-line distributed diagnosis of dynamic systems.
2. A design approach to implement fully independent distributed diagnosers that guarantees that no central coordinator or on-line communication between the local diagnosers is necessary to provide correct diagnosis results.
3. A generalization of PCs to multi-output PCs that merges PCs to compute multi-output PCs, which are necessary to accomplish global diagnosability of the local diagnosers.
4. The development of a distributed fault identification approach, where the PCs are used to compute local parameter estimators.
5. The application of our distributed diagnosis approach to a simulation model of a four-wheeled rover testbed at NASA Ames Research Center [16, 17], to demonstrate the improved design of the proposed distributed solution. Using structural model decomposition, we compute local diagnosers of smaller size compared to the global system model. This improved design is then used to build local event-based diagnosers to demonstrate the diagnosis capabilities of this new distributed diagnosis framework. Results generated for different fault scenarios show that the proposed distributed diagnosis framework generates equivalent fault detection and isolation results to those obtained by the centralized approach. For fault identification, we use the local parameter estimators, which results in an accuracy improvement with respect to the centralized approach.

The paper is organized as follows. Section 2 describes the background, with the system modeling methodology, the diagnosis context, and the theoretical concepts of residual generation, qualitative fault isolation, and event-based diagnosis. Section 3 formulates the distributed diagnosis design problem. Section 4 presents the PCs model decomposition approach and the proposal to generate multi-output PCs. Section 5 describes the event-based distributed diagnosis architecture. Section 6 proposes the local diagnoser design approach. Section 7 describes the methodology to construct the event-based diagnosers, and Section 8 presents the approach for distributed fault identification.

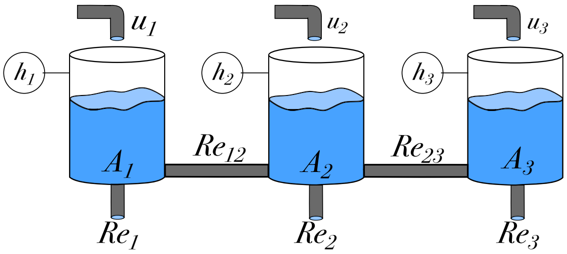


Figure 1: Three-tank system schematic.

Section 9 presents the rover system case study, demonstrating the validity of the approach for different diagnoser designs, and showing results obtained with a number of simulated fault scenarios. Section 10 discusses related work in distributed diagnosis, discrete-event systems, and model decomposition approaches. Finally, Section 11 concludes the paper.

2. Background

In this work we propose a new framework for distributed diagnosis of dynamic systems within the Artificial Intelligence approach for model-based diagnosis (MBD) [3]. In MBD, diagnosis is carried out by comparing measurements in the system against the estimation of the behavior of the system computed by using a *model* of the system. In this section, we first present our system modeling approach, then, we introduce the diagnosis context and the assumptions of this work, and finally we present our qualitative framework for fault isolation.

2.1. System Modeling

In our approach, a model is defined as follows [18]:

Definition 1 (Model)**.** A *model* $\mathcal{M}^*$ is a tuple $\mathcal{M}^* = (V, C)$, where V is a set of variables, and C is a set of constraints among variables in V. V consists of four disjoint sets, namely, the set of state variables, X; the set of parameters, Θ; the set of inputs, U; and the set of outputs, Y. Each constraint $c = (\varepsilon_c, V_c)$, such that $c \in C$, consists of an equation ε_c involving variables $V_c \subseteq V$.

The set of output variables, Y, correspond to the (measured) sensor signals. Parameters, Θ, include explicit model parameters that are used in the model constraints. Regarding input variables, we make the following assumption:

Assumption 1. Input or exogenous variables, U, are known.

Throughout this paper, we will use a standard three tank-system (Fig. 1) as a running example to illustrate the basic concepts, formulate the problem, and explain our primary definitions. The tanks are connected serially, and the model of the system is represented by the following set of constraints:[3]

$$p_1 = \frac{\rho \cdot g}{A_1} \int_{t_0}^{t} \Big(u_1 - \frac{1}{Re_1}(p_1) - \frac{1}{Re_{12}}(p_1 - p_2)\Big) \cdot dt \tag{c_1}$$

$$p_2 = \frac{\rho \cdot g}{A_2} \int_{t_0}^{t} \Big(u_2 + \frac{1}{Re_{12}}(p_1 - p_2) - \frac{1}{Re_2}(p_2) - \frac{1}{Re_{23}}(p_2 - p_3)\Big) \cdot dt \tag{c_2}$$

$$p_3 = \frac{\rho \cdot g}{A_3} \int_{t_0}^{t} \Big(u_3 + \frac{1}{Re_{23}}(p_2 - p_3) - \frac{1}{Re_3}(p_3)\Big) \cdot dt \tag{c_3}$$

$$h_1 = \frac{p_1}{\rho \cdot g} \tag{c_4}$$

$$h_2 = \frac{p_2}{\rho \cdot g} \tag{c_5}$$

$$h_3 = \frac{p_3}{\rho \cdot g} \tag{c_6}$$

where, for tank i, u_i denotes the input flow, p_i denotes the pressure in the tank, A_i denotes the cross section of the tank, and Re_i denotes the resistance of the connected drain pipe. For tanks i and j, Re_{ij} denotes the connecting pipe resistance. Three output sensors, h_1, h_2, and h_3, measure the level in the three tanks.

Example 1. For the three-tank system, the model $\mathcal{M}^*$ is represented by the variable sets $X = \{p_1, p_2, p_3\}$, $\Theta = \{A_1, A_2, A_3, Re_1, Re_2, Re_3, Re_{12}, Re_{23}\}$, $U = \{u_1, u_2, u_3\}$, and $Y = \{h_1, h_2, h_3\}$; and the set of constraints $C = \{c_1, c_2, c_3, c_4, c_5, c_6\}$.

The notion of a *causal assignment* is used to specify the computational causality for a constraint c, by defining which $v \in V_c$ is the dependent variable in equation ε_c.

Definition 2 (Causal Assignment)**.** A *causal assignment* α to a constraint $c = (\varepsilon_c, V_c)$ is a tuple $\alpha = (c, v_c^{out})$, where $v_c^{out} \in V_c$ is assigned as the dependent variable in ε_c.

We write a causal assignment of a constraint using its equation in a causal form, with := to explicitly denote the causal (i.e., computational) direction.

Definition 3 (Valid Causal Assignments)**.** We say that a set of causal assignments $\mathcal{A}$, for a model $\mathcal{M}^*$ is *valid* if

- For all $v \in U \cup \Theta$, $\mathcal{A}$ does not contain any α such that $\alpha = (c, v)$.
- For all $v \in Y$, $\mathcal{A}$ does not contain any $\alpha = (c, v_c^{out})$ where $v \in V_c - \{v_c^{out}\}$.
- For all $v \in V - U - \Theta$, $\mathcal{A}$ contains exactly one $\alpha = (c, v)$.

The definition of valid causal assignments implies that input or parameter variables cannot be the dependent variables in the causal assignment, a measured variable can be used as the dependent variable, and every variable, which is not input or parameter, is only computed by one (causal) constraint. Based on this, a *causal model* is a model extended with a valid set of causal assignments.

Definition 4 (Causal Model)**.** Given a model $\mathcal{M}^* = (V, C)$, a *causal model* for $\mathcal{M}^*$ is a tuple $\mathcal{M} = (V, C, \mathcal{A})$, where $\mathcal{A}$ is a set of valid causal assignments.

Example 2. For the three-tank system model, the causal constraints are as follows.

[3] Since we are working with dynamic systems, the dynamic behavior can be modeled using either integration or differentiation [11, 12, 14, 19]. In this work, we have used integration to avoid problems of differentiation regarding noisy measurements and others. However, this is just an implementation decision, and the approach presented in this work is independent of how dynamic behavior is modeled.

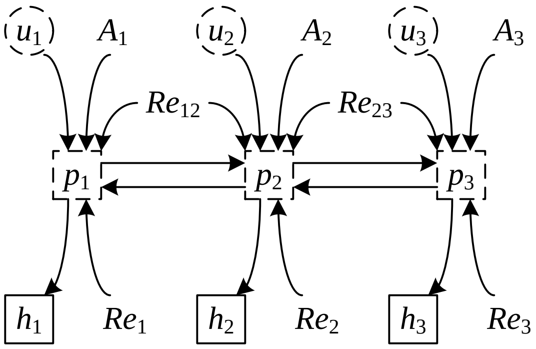


Figure 2: Causal graph for the three-tank system model.

$$p_1 := \frac{\rho \cdot g}{A_1} \int_{t_0}^{t} \Big(u_1 - \frac{1}{Re_1}(p_1) - \frac{1}{Re_{12}}(p_1 - p_2)\Big) \cdot dt \tag{α_1}$$

$$p_2 := \frac{\rho \cdot g}{A_2} \int_{t_0}^{t} \Big(u_2 + \frac{1}{Re_{12}}(p_1 - p_2) - \frac{1}{Re_2}(p_2) - \frac{1}{Re_{23}}(p_2 - p_3)\Big) \cdot dt \tag{α_2}$$

$$p_3 := \frac{\rho \cdot g}{A_3} \int_{t_0}^{t} \Big(u_3 + \frac{1}{Re_{23}}(p_2 - p_3) - \frac{1}{Re_3}(p_3)\Big) \cdot dt \tag{α_3}$$

$$h_1 := \frac{p_1}{\rho \cdot g} \tag{α_4}$$

$$h_2 := \frac{p_2}{\rho \cdot g} \tag{α_5}$$

$$h_3 := \frac{p_3}{\rho \cdot g} \tag{α_6}$$

The causal model $\mathcal{M}$ is represented by the variable sets $X = \{p_1, p_2, p_3\}$, $\Theta = \{A_1, A_2, A_3, Re_1, Re_2, Re_3, Re_{12}, Re_{23}\}$, $U = \{u_1, u_2, u_3\}$, and $Y = \{h_1, h_2, h_3\}$; the set of constraints $C = \{c_1, c_2, c_3, c_4, c_5, c_6\}$; and the set of causal assignments $\mathcal{A} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6\}$.

We can visualize a causal model $\mathcal{M}$ using a directed graph $\mathcal{G} = (N, A)$, where N is the set of nodes corresponding directly to the variables V in $\mathcal{M}$, and A is the set of arcs, where for every $(c, v_c^{out}) \in \mathcal{A}$, we include an arc (v', v_c^{out}) for each $v' \in V_c - \{v_c^{out}\}$.

Example 3. The causal graph corresponding to the three-tank system model is given in Fig. 2. In the graph, we mark inputs with dashed circles, state variables with dashed squares, and outputs with solid squares.

2.2. *Consistency-based Diagnosis*

As we previously mentioned, in this work we will approach the fault diagnosis problem using model-based diagnosis. More precisely, our proposal is based on the consistency-based diagnosis (CBD) approach as proposed by Reiter in [2]. But, we extend the approach to dynamic and distributed systems using concepts from the Control Engineering approach to MBD, known as the FDI approach, given the ongoing interest in both communities to develop a common framework for MBD [20–23].

As pointed out in several reviews of the CBD field [5, 21], there is no general theoretical framework for CBD of dynamic systems [24], but a collection of several works following the same consistency-based principles for diagnosis using different kind of models and different temporal ontologies exists [5, 7, 25–27]. As a consequence, we will start from the basic principles stated by de Kleer and Kurien [21] for static systems, and we will state the relevant working hypotheses [5] in our diagnosis process for dynamic systems:

- Our system model is equation-based, not component-based. That is, it is made up of a set of equations that contains variables and/or parameters. Changes in some of these parameters will be responsible for the faulty behavior of the system. In many engineered systems it is possible to go from a equation-based to a component-based model, thus providing a higher level of abstraction in the results.
- We assume that the presence of faults will not introduce "genuine" new equations in the system, i.e. will not change the model structure.[4]
- The system model describes the behavior of the system, not how to diagnose it (e.g., as it happens in expert systems).
- A domain-independent reasoning engine performs the diagnosis task using the model.
- We assume that the set of measurements are fixed, and all of the diagnostic reasoning is based on the equation-based model and the values observed for the measurements. No additional measurements and model information becomes available once the diagnosis is initiated.[5]
- Finally, we do not impose that all of the computations required to generate diagnostic candidates is done online. Information may be pre-computed from the model. But this assumption can be relaxed.[6]

MBD proceeds in an iterative process of fault detection, fault isolation (or localization), and fault identification. In our context, a fault is the cause of an unexpected, persistent deviation of the system behavior from the acceptable nominal behavior. Moreover, since our proposal is equation-based and not component-based, we will link fault candidates to the set of parameters Θ in $\mathcal{M}$.

In CBD, fault detection is carried out based on conflicts, that are related to a set of correctness assumptions for the model components that are not consistent with current measured values in the system. In this work, we associate the correct behavior assumptions to the nominal value of the parameters in Θ [2, 28],[7] and conflicts are computed as subsets of the elements $\theta \in \Theta$.[8]

The set of conflicts can be computed on-line using an ATMS-like engine [28, 31], propagating backward in a causal model [7, 26], or can be precomputed using off-line dependency-recording techniques such as Possible Conflicts [14], Potential Conflicts [23], or equivalent techniques. It should be noted that the works by Cordier et al. [23] and de Kleer and Kurien [21] stated the similarities between Reiter's theory for diagnosis and the FDI approach based on ARRs. In that work, the support for an ARR was the set of correctness assumption linked to components that were involved in an ARR, and they were considered as equivalent to the correctness assumptions in a conflict. In an equation-based approach the parameters play the role of components, with respect to correctness assumptions. In that way we can redefine a conflict as:

Definition 5 (Conflict)**.** A conflict is the set of correctness assumptions, $\theta \subseteq \Theta$, related to the equations required to estimate a discrepancy.

In our framework, a conflict will arise from the discrepancy between a measurement and an estimation of the measurement. Fault isolation is straightforward in CBD once we have conflicts. We only need to compute the minimal hitting sets of the conflicts, thus leading to faulty parameters that are inconsistent with the whole set of detected discrepancies.

In our approach, we go a step further because our model is able to predict faulty behavior too. Thus we can provide a more precise set of candidates if we take into account how faults are related to parameters [11, 32]:

[4] This way we also eliminate the possibility to isolate *bridge* or structural faults, as referred in early DX literature.

[5] In most on-line real systems the set of measurements is known and fixed. We are monitoring the system on-line, based on this set of measurements, and it is from this on-line monitoring process where we determine that something is wrong and we perform our diagnosis. If we work on a post-mortem diagnosis scenario, additional measurement points could be considered. But this is not our case.

[6] We think that this is not a major difference with the original approach, and we could use a pure on-line model-based diagnoser in the implementation stage.

[7] These correctness assumptions are modeled as $OK(\cdot)$ or $\neg AB(\cdot)$ in [2] and [28], respectively.

[8] Similarly, this is the assumption used in the structural approaches to MBD from the FDI field to compute the structure of Analytical Redundancy Relations, ARRs. These approaches have been demonstrated to be equivalent to conflict calculation for both static [23, 29], and dynamic systems [30].

Definition 6 (Fault)**.** A *fault* is a deviation of exactly one parameter of the system model from its nominal value.

A fault is denoted as f, and it is modeled, in our work, as an unexpected step change in a system model parameter value, $\theta \in \Theta$, and F represents the fault set. Faults are named by the associated parameter and the direction of change, i.e., θ^+ (resp., θ^-) denotes a fault defined as an abrupt increase (resp., decrease) in the value of parameter θ.

Example 4. In the three-tank system in Fig. 1, the fault set considered is $F = \{A_1^-, A_2^-, A_3^-, Re_1^+, Re_2^+, Re_3^+, Re_{12}^+, Re_{23}^+\}$.

We make the following assumption regarding faults:

Assumption 2. Only single faults occur in the system.[9]

Since our main goal is the accurate identification of a single fault, we can define *diagnosis* in this context as:

Definition 7 (Diagnosis)**.** A *diagnosis* is a set of single fault hypotheses that are consistent with the observations.

In this work we will not use the classical GDE approach to consistency-based diagnosis extended for dynamic systems. We will use a structural approach to determine off-line the submodels that can generate a discrepancy, as will be explained in Section 4. These submodels will contain a set of correctness assumptions that can be identified as potential conflicts [14, 23]. Once the submodels are computed off-line, we build an executable model that will be responsible for behavior estimation. Whenever we find a discrepancy between observed and estimated behavior we consider that the potential or possible conflict is present, and we can obtain the set of fault candidates by computing the minimal hitting-set for the sets of correctness assumptions.

Since we work with dynamic systems and a given set of measurements, in order to isolate fault diagnosis candidates we can proceed in different ways: waiting for more observations to deviate, using qualitative information about the system measurements, or using predictive fault models. In our approach, we will use the three options within a CBD approach with fault models, where we will reject inconsistent fault mode assignments, as we describe next.

2.3. Qualitative Framework for Fault Isolation

In this subsection, we recapitulate the basic theoretical concepts that our diagnosis approach is based on. We first present the concept of residual, review the theoretical framework for qualitative fault isolation [7], and then review the framework for event-based fault modeling [9].

Faults in the system manifest as persistent abrupt changes in the value of the system parameters, causing transients in the system variables that are observed as deviations of measured values from predicted values, triggering residuals. The evolution of these deviations caused in the residuals are used to isolate the true faults. A *residual* is an equivalent concept to discrepancy, used by researchers from both the Artificial Intelligence and the Control Engineering fields, and it is defined as follows:

Definition 8 (Residual)**.** A *residual*, r_y, is a time-varying signal computed as the difference between a measurement, $y \subseteq Y$, and a predicted value of the measurement y, denoted as $\hat{y}$.

Concerning fault diagnosis, a residual is a fault indicator, based on an observed deviation between measurements and model-based estimations. The predicted value of the measurement is computed by using the model of the system [7] (this model can be either the global model of the system or a submodel of this global model with enough redundancy to estimate the measured variable y). Hence, the residual is ideally zero in the nominal situation, and nonzero when a fault occurs in the system. The residual set is denoted as R.

Example 5. In the three-tank system example, since we measure the level in each one of the tanks, we can compute three different residuals, r_{h_1}, r_{h_2}, and r_{h_3}, where r_{h_1} (cf., r_{h_2} and r_{h_3}) is computed as the difference between the measurement of the level in tank 1 (cf., 2 and 3) and the prediction of the level in tank 1 (cf., 2 and 3).

[9] This assumption implies that all of the observed deviations in the measurements can be explained by a single deviated parameter value.

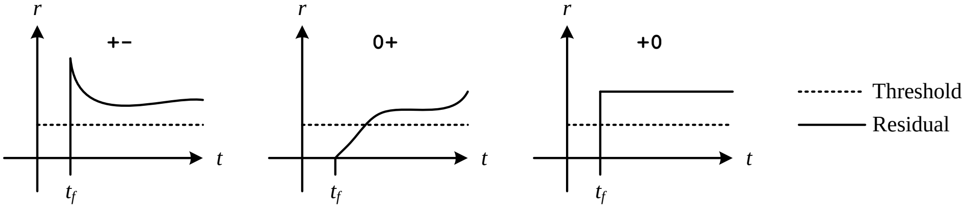


Figure 3: Residual deviations and corresponding fault signatures.

When faults occur, they produce deviations in the residuals from zero, and it is this information that is used to generate conflicts and isolate faults. Because reasoning over the continuous residual signals is difficult and computationally demanding, we abstract a residual into a symbolic form. The transient in the residual signal at the time it is triggered is abstracted using qualitative +, −, and 0 values in the signal magnitude and derivatives, justified by the Taylor series expansion of the signal [33]. Consequently, the interpretation for these qualitative values is: a "0" means the observation is within the nominal thresholds, i.e., $\hat{y}-T < y < \hat{y}+T$; a "+" simply means the observation y is above the predicted output $\hat{y}$ plus the threshold T, i.e., $y > \hat{y} + T$; and a "-" means the observation is below the predicted output minus the threshold, i.e., $y < \hat{y} - T$. We retain only the change in magnitude and the first nonzero derivative change, which, over time, will manifest as a change in slope.

Definition 9 (Fault Signature)**.** A *fault signature* for a fault f and residual r, denoted by $\sigma_{f,r}$, is pair of symbols $s_1 s_2$ representing potential qualitative changes in magnitude and slope of r caused by f at the point of the occurrence of f. The set of fault signatures for f and r is denoted as $\Sigma_{f,r}$.

Note that this definition of a fault signature is independent of the system inputs and fault magnitude. If, for example, for f and r, the fault signature set $\Sigma_{f,r} = \{+-\}$, this means that the fault will always produce an immediate increase in residual magnitude followed by a decrease in residual slope. $\Sigma_{f,r}$ is usually a singleton because signatures are defined with respect to the residual, which is a difference. An observed fault signature on residual r_y for measurement y is written as $r_y^{s_1 s_2}$, e.g., $r_{h_1}^{+-}$.

The complete set of possible fault signatures for a residual we consider here is $\{+-, -+, 0+, 0-, +0, -0\}$. Residual deviations in the positive direction and their corresponding signatures are shown in Fig. 3.

Fault signatures form the basis for fault isolation. We use both the qualitative values of the fault signature and the temporal order of the residual deviations as discriminatory information. The temporal order of residual deviations for a given model, termed *relative residual orderings* [34], are based on the intuition that fault effects will manifest in some parts of the system before others. As previously described in this section, for a given model, there is a residual defined for each measurement in the model. Within this model, the relative ordering of the residual deviations can be computed based on analysis of the transfer functions from faults to residuals, as proven in [34].

Definition 10 (Relative Residual Ordering)**.** If fault f always manifests in residual r_i before residual r_j, then we define a *relative residual ordering* between r_i and r_j for fault f, denoted by $r_i \prec_f r_j$. We denote the set of all residual orderings for f as $\Omega_{f,R}$.

Note that in this definition, we are referring specifically to deviations in the residuals caused by faults. In this paper, to make the approach as general as possible, we assume that fault signatures and relative residual orderings are given as inputs. In practice, this information can be generated by manual analysis of the system model, by simulation, or automatically from certain types of models, e.g., as presented in [35].

Example 6. Table 1 shows the predicted fault signatures and residual orderings for the model of a three-tank system with $F = \{A_1^-, A_2^-, A_3^-, Re_1^+, Re_2^+, Re_3^+, Re_{12}^+, Re_{23}^+\}$, $Y = \{h_1, h_2, h_3\}$, and $R = \{r_{h_1}, r_{h_2}, r_{h_3}\}$. For example, consider A_1^-. An abrupt decrease in A_1^- would cause an abrupt increase in p_1 (see c_1), and an abrupt increase in h_1 (see c_4). The increase in p_1 would cause an increase in the flow to the second tank, which through the integration manifests as a first-order increase in p_2 and h_2 (resulting in $r_{h_2}^{0+}$). Similarly the increase in p_2 causes a second-order increase in p_3

and h_3 (resulting in $r_{h_3}^{0+}$). The first-order increase in p_2 also causes a second-order decrease in p_1 and h_1, resulting in $r_{h_1}^{+-}$. Because of the integrations, the abrupt change in r_{h_1} is observed first, followed by the change in r_{h_2} and then r_{h_3}, resulting in the residual orderings $r_{h_1} \prec r_{h_2}$, $r_{h_1} \prec r_{h_3}$, and $r_{h_2} \prec r_{h_3}$.

Table 1: Fault signatures and relative residual orderings for the global model, $\mathcal{M}$, of the three-tank system.

Fault	r_{h_1}	r_{h_2}	r_{h_3}	Residual Orderings
A_1^-	$+-$	$0+$	$0+$	$r_{h_1} \prec r_{h_2}, r_{h_1} \prec r_{h_3}, r_{h_2} \prec r_{h_3}$
Re_1^+	$0+$	$0+$	$0+$	$r_{h_1} \prec r_{h_2}, r_{h_1} \prec r_{h_3}, r_{h_2} \prec r_{h_3}$
Re_{12}^+	$0+$	$0-$	$0-$	$r_{h_2} \prec r_{h_3}$
A_2^-	$0+$	$+-$	$0+$	$r_{h_2} \prec r_{h_1}, r_{h_2} \prec r_{h_3}$
Re_2^+	$0+$	$0+$	$0+$	$r_{h_2} \prec r_{h_1}, r_{h_2} \prec r_{h_3}$
Re_{23}^+	$0+$	$0+$	$0-$	$r_{h_2} \prec r_{h_1}$
A_3^-	$0+$	$0+$	$+-$	$r_{h_2} \prec r_{h_1}, r_{h_3} \prec r_{h_1}, r_{h_3} \prec r_{h_2}$
Re_3^+	$0+$	$0+$	$0+$	$r_{h_2} \prec r_{h_1}, r_{h_3} \prec r_{h_1}, r_{h_3} \prec r_{h_2}$

Together, fault signatures and relative residual orderings establish an event-based form of diagnostic information. For a given fault, the combination of all fault signatures and residual orderings yields all the possible ways a fault can manifest in the residuals. We define each of these possibilities as a *fault trace*.

Definition 11 (Fault Trace)**.** A *fault trace* for a fault f over residuals R, denoted by $\lambda_{f,R}$, is a sequence of fault signatures, of length $\leq |R|$ that includes, for every $r \in R$ that will deviate due to f, a fault signature $\sigma_{f,r}$, such that the sequence of fault signatures satisfies $\Omega_{f,R}$.

Note that fault traces are required to be maximal in such way that a fault signature is included in the trace for every residual that will deviate due to the fault.

Example 7. Given $R = \{r_{h_1}, r_{h_2}, r_{h_3}\}$, for fault A_2^-, from Table 1 we see that the fault effects will appear first on r_{h_2}, and then it is unknown whether r_{h_1} or r_{h_3} will deviate next. Hence, there are two possible fault traces: $r_{h_2}^{+-} r_{h_1}^{0+} r_{h_3}^{0+}$ and $r_{h_2}^{+-} r_{h_3}^{0+} r_{h_1}^{0+}$. On the other hand, for A_3^-, there is only one possible fault trace, $r_{h_3}^{+-} r_{h_2}^{0+} r_{h_1}^{0+}$.

We group the set of all fault traces into a *fault language*. The *fault model*, defined by a finite automaton, concisely represents the fault language of a fault.

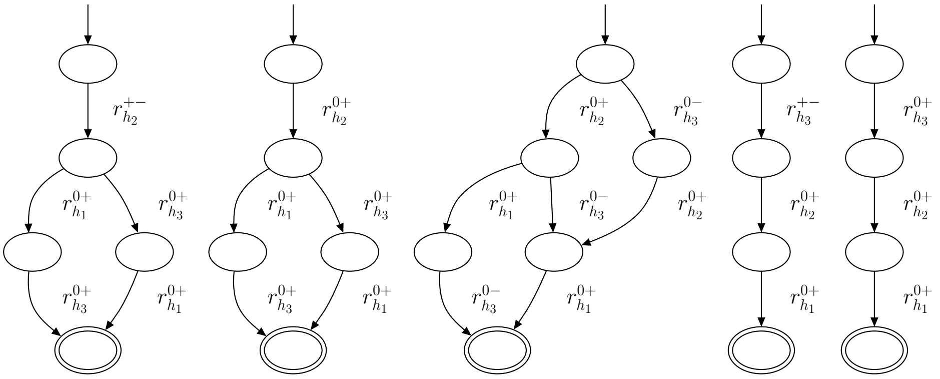
Definition 12 (Fault Language)**.** The *fault language* of a fault $f \in F$ with residual set R, denoted by $L_{f,R}$, is the set of all fault traces for f over the residuals in R.

Definition 13 (Fault Model)**.** The *fault model* for a fault $f \in F$ with residual set R, is the finite automaton that accepts exactly the language $L_{f,R}$, and is given by $\mathcal{L}_{f,R} = (S, s_0, \Sigma, \delta, A)$ where S is a set of states, $s_0 \in S$ is an initial state, Σ is a set of events, $\delta : S \times \Sigma \to S$ is a transition function, and $A \subseteq S$ is a set of accepting states.

Example 8. Selected fault models for the three-tank system are shown in Fig. 4. For example, as seen in $\mathcal{L}_{A_2^-,R}$, the fault A_2^- may manifest as the fault traces $r_{h_2}^{+-} r_{h_1}^{0+} r_{h_3}^{0+}$ or $r_{h_2}^{+-} r_{h_3}^{0+} r_{h_1}^{0+}$, as implied by the fault signatures and residual orderings.

From the fault models, it is clear that each fault may manifest in the residuals in several ways. This ambiguity arises from two sources. First, there may be more than one fault signature for a given fault and residual. Second, the residual orderings, in general, define only a partial ordering of the residual deviations. As a result, there are several potential fault traces associated with each fault.

In this framework, our set of correctness assumptions state that each potential faulty parameter in the model is initially at a nominal value. In on-line diagnosis, until a residual is observed to deviate, the minimal diagnosis is the empty diagnosis, $\varnothing$, since each (observable) fault is predicted to generate at least one residual deviation. When the first residual is observed to deviate, we observe the fault signature through signal processing methods (as it is explained



(a) $\mathcal{L}_{A_2^-,R}$ (b) $\mathcal{L}_{Re_2^+,R}$ (c) $\mathcal{L}_{Re_{23}^+,R}$ (d) $\mathcal{L}_{A_3^-,R}$ (e) $\mathcal{L}_{Re_3^+,R}$

Figure 4: Fault models for some faults of the three-tank system, where $R = \{r_{h_1}, r_{h_2}, r_{h_3}\}$.

in Section 5. We then generate a conflict and we perform fault isolation through the sets of faults whose signatures are consistent with observed residual deviations. In CBD, the first diagnosis set is derived as the minimal hitting set of the current diagnosis set, $\emptyset$, with the conflict, so the new diagnosis set is the conflict. From this point on, since we make the single fault assumption (Assumption 2), the new diagnosis set is computed simply as the intersection of the current diagnosis set and the new conflict. When the next residual is observed to deviate, we generate the next conflict following the same procedure. Formally, for an observed fault signature σ_i on residual r, with $\lambda_{i-1}\sigma_i$ being the observed fault trace up to this point, the fault candidates are generated as the set of faults where for f, there exists a trace $\lambda_{f,R} \in L_{f,R}$ such that $\lambda_{i-1}\sigma_i \sqsubseteq \lambda_{f,R}$. The new diagnosis set is computed as the minimal hitting set of the conflict and the previous diagnosis. As new fault signatures are observed, the process continues.

Example 9. Consider the set of faults $\{A_2^-, Re_2^+, Re_{23}^+, A_3^-, Re_3^+\}$, and the set of residuals $\{r_{h_1}, r_{h_2}, r_{h_3}\}$ (see Fig. 4). Say that the first observed fault signature is $r_{h_2}^{0+}$. Then the conflict is $\{Re_2^+, Re_{23}^+\}$, and the new diagnosis set is $\{Re_2^+, Re_{23}^+\}$. Say the next signature is $r_{h_3}^{0+}$. The conflict is $\{Re_2^+\}$, as only that fault can produce that $r_{h_2}^{0+} r_{h_3}^{0+}$ as the beginning of a fault trace. The new diagnosis set is then $\{Re_2^+\}$, and a unique diagnosis is achieved.

Although our diagnosis framework will make an extensive use of qualitative information about residuals, it should be pointed out that we need to use quantitative models for behavior estimation, because a precise estimation will allow our diagnoser to perform quick fault detection, and will be the input for quantitative fault identification later. This issue will be further discussed in Section 5.

3. Problem Formulation

As mentioned in Section 1, consistency-based diagnosis frameworks for continuous systems are typically implemented using a centralized approach. This centralized approach is based on the global model of the system. However, centralized approaches face several drawbacks: they are expensive in memory and computation, do not scale well as the size of the system grows [4–6], and also introduce a single point of failure.

Distributed diagnosis approaches break down the diagnosis problem into different subtasks that may be executed on separate processors, however a central coordinator is typically needed in distributed solutions to form a consistent global diagnosis from the local diagnoses [36, 37]. To overcome such a drawback, we propose a distributed diagnosis approach capable of designing local diagnosers that generate globally correct local diagnosis results, with no central coordinator.

Diagnoser design uses the notion of global diagnosability, i.e., every local diagnoser must produce globally correct results [8, 10]. Given a model of a system, and the set of faults (F) and residuals (R), we may establish the notions of *distinguishability* and *diagnosability*. Using these definitions, we can formally define the globally diagnosable diagnosis submodel design problem.

In general, two faults are distinguishable if they always, in finite time, produce different residuals. In our diagnosis framework, distinguishability between faults is characterized using fault traces and languages.

Definition 14 (Distinguishability)**.** Given a residual set, R, a fault f_i is *distinguishable* from a fault f_j, denoted by $f_i \not\sim_R f_j$, if there does not exist a pair of fault traces $\lambda_{f_i,R} \in L_{f_i,R}$ and $\lambda_{f_j,R} \in L_{f_j,R}$, such that $\lambda_{f_i} \sqsubseteq \lambda_{f_j}$.

One fault will be distinguishable from another fault if it cannot produce a fault trace that is a prefix[10] (denoted by $\sqsubseteq$) of a trace that can be produced by the other fault. If this is not the case, then when that trace manifests, the first fault cannot be distinguished from the second. This implementation of distinguishability should be clear from the description of how conflicts are generated and diagnoses are computed in Section 2.3. Consider the following example.

Example 10. Consider the the faults Re_2^+ and Re_{23}^+, and the set of residuals $\{r_{h_1}, r_{h_2}, r_{h_3}\}$ (see Table 1 and Fig. 4). Re_2^+ is distinguishable from Re_3^+ because if Re_3^+ occurs, it will produce either $r_{h_2}^{0+} r_{h_3}^{0+} r_{h_1}^{0+}$ or $r_{h_2}^{0+} r_{h_1}^{0+} r_{h_3}^{0+}$, neither of which Re_{23}^+ can produce. Consider now the same faults but residuals $\{r_{h_1}, r_{h_2}\}$. Re_2^+ will produce only $r_{h_2}^{0+} r_{h_1}^{0+}$, which can also be produced by Re_{23}^+, therefore, with this set of residuals Re_2^+ is not distinguishable from Re_{23}^+. Similarly, Re_{23}^+ is not distinguishable from Re_2^+ because it can produce $r_{h_2}^{0+} r_{h_1}^{0+}$, which Re_2^+ can also produce.

Distinguishability requires that in all circumstances two faults cannot produce the same observable behavior. For example, if a fault f_1 can produce a trace that another fault f_2 cannot, but can also produce a trace that f_2 can, we still say they are not distinguishable. In practice, if the second trace is the one that manifests, we will distinguish them, however, since we are tackling a design problem in which we want to guarantee distinguishability, we must take into account that either trace can be produced, so we cannot claim that the faults will always be distinguished.

Distinguishability is used to define the diagnosability of a diagnosis model under a given fault isolation framework. A diagnosis model is an abstraction of a system model with only diagnosis relevant information, and it is defined as follows.

Definition 15 (Diagnosis Model)**.** A *diagnosis model* $\mathcal{S}$ is a tuple $(F, Y, R, L_{F,R})$, where $F = \{f_1, f_2, \ldots, f_n\}$ is a set of faults, Y is a set of measurements, R is a set of residuals, and $L_{F,R} = \{L_{f_1,R}, L_{f_2,R}, \ldots, L_{f_n,R}\}$ is the set of fault languages.

Example 11. A diagnosis model $\mathcal{S}$ for the three-tank system is represented by the variable sets $F = \{A_1^-, A_2^-, A_3^-, Re_1^+,$ Re_2^+, Re_3^+, Re_{12}^+, $Re_{23}^+\}$, $Y = \{h_1,\ h_2,\ h_3\}$, $R = \{r_{h_1},\ r_{h_2},\ r_{h_3}\}$, and the set of fault languages $L_{F,R} = \{L_{A_1^-,R}$, $L_{A_2^-,R}$, $L_{A_3^-,R}$, $L_{Re_1^+,R}$, $L_{Re_2^+,R}$, $L_{Re_3^+,R}$, $L_{Re_{12}^+,R}$, $L_{Re_{23}^+,R}\}$

If a diagnosis model is diagnosable, then we can guarantee about the unique isolation of every fault in the diagnosis model.

Definition 16 (Diagnosability)**.** A diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$ is *diagnosable* if and only if $(\forall f_i,\ f_j \in F) f_i \neq f_j \implies f_i \not\sim_R f_j$.

If $\mathcal{S}$ is diagnosable, then every pair of faults is distinguishable using the residual set R. Hence, we can uniquely isolate all faults of interest. If $\mathcal{S}$ is not diagnosable, then ambiguities will remain after fault isolation, i.e., after all possible fault effects on the residuals have been observed.

Example 12. Consider the $\mathcal{M}$-based residual set given in Table 1. A diagnosis model defined with these residuals is diagnosable when both fault signatures and residual orderings are used (without orderings, faults Re_1^+, Re_2^+, and Re_3^+ have all the same signatures and cannot be distinguished from one another).

Our objective is to decompose the overall diagnosis task into smaller subtasks performed by local diagnosers with the following properties: *(i)* all single faults of interest in the diagnosis model can be diagnosed, and *(ii)* the local diagnosis results are *globally* correct. These two properties eliminate the need for a centralized coordinator. In order to decompose the diagnosis task based on diagnosability, the diagnosis model must be diagnosable, hence we have the following assumption.

[10] A fault trace λ_i is a prefix of fault trace λ_j if there is some (possibly empty) sequence of events λ_k that can extend λ_i such that $\lambda_i \lambda_k = \lambda_j$.

Assumption 3. The global system is always diagnosable for the $\mathcal{M}$-based residual set.

However, if the diagnosis model is not diagnosable, we can define *aggregate faults*, where an aggregate fault is a set of faults that are indistinguishable from each other. Our diagnosis approach can be applied to the modified fault set that includes the aggregate faults. If an aggregate fault is diagnosed, that implies that one of its constituent faults has occurred.

The diagnosis model $\mathcal{S}$ is split into n diagnosis submodels $\mathcal{S}_1$, $\mathcal{S}_2$, ..., $\mathcal{S}_n$, where each diagnosis submodel gets a subset of the fault set, a subset of the measurement set, and a subset of the residual set. The desired distribution of faults and measurements within each diagnosis submodel in initially provided by the user as input, e.g., based on subsystems of the system.

Definition 17 (Diagnosis Submodel)**.** A *diagnosis submodel* $\mathcal{S}_i$ from a diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$ is a tuple $(F_i, Y_i, R_i, L_{F_i,R_i})$, where $F_i \subseteq F$, $Y_i \subseteq Y$, and $R_i \subseteq R$.

We are interested in a set of diagnosis submodels having enough local diagnostic information to get the correct global answer. This is referred to as *global correctness*, which says that for any fault that occurs, the union of indistinguishable faults over all diagnosis submodels will be that fault itself.

Definition 18 (Global Correctness)**.** A set of n diagnosis submodels $\{\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i}) : i = 1, \ldots, n\}$ from a diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$ is *globally correct* if for all $f \in F$, $\bigcup_{\mathcal{S}_i}\{f_i \in F_i : f_i \sim_{R_i} f\} = \{f\}$.

Note that global correctness enforces the constraint that the set of diagnosis submodels covers all faults, i.e., that each fault $f \in F$ is contained in at least one F_i. To keep local diagnosers as small as possible, we will enforce a stricter constraint that each fault is contained in exactly one F_i, so that every fault has exactly one diagnosis submodel that is responsible for it.

Diagnosis submodels may be locally diagnosable. A locally diagnosable diagnosis submodel is one in which its own faults can be uniquely isolated using its own residuals.

Definition 19 (Local Diagnosability)**.** A diagnosis submodel $\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i})$ is *locally diagnosable* if $(\forall f_i, f_j \in F_i)\ f_i \neq f_j \implies f_i \nsim_{R_i} f_j$. We say fault $f_i \in F_i$ is *locally distinguishable* from $f_j \in F_i$ if $f_i \nsim_{R_i} f_j$.

Note that the definition of local diagnosability for a diagnosis submodel is equivalent to the definition of diagnosability for a diagnosis model (Definition 16). For a set of diagnosis submodels, local diagnosability is not a strong enough condition to ensure global correctness. The problem is that for two different diagnosis submodels, $\mathcal{S}_i$ and $\mathcal{S}_j$, there may be some faults $f_i \in F_i$ and $f_j \in F_j$, such that both f_i and f_j produce the same effects on R_i. Hence, if fault f_j occurs in the system, the local diagnoser for $\mathcal{S}_i$ will think that fault f_i has occurred, which is not globally correct, i.e., we may have faults that are distinguishable from faults *inside* the diagnosis submodel, but not distinguishable from faults *outside* the diagnosis submodel.

Example 13. Consider a diagnosis submodel $\mathcal{S}_i$ where $F_i = \{A_1^-, Re_1^+\}$, $Y_i = \{h_1\}$, and $R_i = \{r_{h_1}\}$. This diagnosis submodel is locally diagnosable, because its two faults produce different effects on its single residual: A_1^- produces $r_{h_1}^{+-}$ and Re_1^+ produces $r_{h_1}^{0+}$ (see Table 1). But every fault in $F - F_i$ can also produce $r_{h_1}^{0+}$, so if one of those faults occurs, the diagnoser for $\mathcal{S}_i$ would isolate Re_1^+ as the fault, which is not globally correct.

Therefore, we require a notion of *global* diagnosability.

Definition 20 (Global Diagnosability)**.** A diagnosis submodel $\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i})$ from diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$ is *globally diagnosable* if $(\forall f_i \in F_i, f_j \in F)\ f_i \neq f_j \implies f_i \nsim_{R_i} f_j$. We say fault $f_i \in F_i$ is *globally distinguishable* from $f_j \in F$ if $f_i \nsim_{R_i} f_j$.

That is, a diagnosis submodel $\mathcal{S}_i$ is *globally diagnosable* if all the faults F_i are distinguishable from every other fault in F, i.e., all other faults in S, using only the residuals in R_i.

Example 14. Consider again the diagnosis submodel $\mathcal{S}_i$ where $F_i = \{A_1^-, Re_1^+\}$, $Y_i = \{h_1\}$, and $R_i = \{r_{h_1}\}$. This diagnosis submodel is not globally diagnosable, because faults not included in F_i can produce the same observations as a fault in F_i, as demonstrated in the previous example. Now consider that $Y_i = \{h_1, h_2\}$, and $R_i = \{r_{h_1}, r_{h_2}\}$. Now Re_1^+ is distinguishable from Re_{12}^+ and A_2^- because the effect on r_{h_2} is different, and Re_1^+ is distinguishable from the remaining faults not in F_i because it produces a change in r_{h_1} before r_{h_2}, whereas the others produce a change in r_{h_2} before r_{h_1}.

If the diagnosis submodels can be structured such that all faults in F are covered and each diagnosis submodel $\mathcal{S}_i$ is globally diagnosable, then each diagnosis submodel will generate globally correct diagnoses without communicating its diagnosis results to other diagnosis submodels. The following lemma shows that global diagnosability ensures the desired quality of global correctness.

Lemma 1. *If each diagnosis submodel in a set of n diagnosis submodels* $\{\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i}) : i = 1, \ldots, n\}$*, from a diagnosis model* $\mathcal{S} = (F, Y, R, L_{F,R})$ *where* $F = F_1 \cup F_2 \cup \ldots \cup F_n$*, is globally diagnosable, then the set of diagnosis submodels is globally correct.*

Proof. Given any fault $f \in F$, because all diagnosis submodels are globally diagnosable, the fault must be distinguishable from every local and non-local fault except for that fault itself. Hence, since f must be contained in at least one diagnosis submodel, the union of all indistinguishable faults from each diagnosis submodel must equal $\{f\}$. Therefore, the set of diagnosis submodels is globally correct. □

In this paper, we focus on the problem where the diagnosis model $\mathcal{S}$ is diagnosable and the user provides the initial distribution of the measurement and fault sets.[11] Usually, each $\mathcal{S}_i$ may not be globally diagnosable. The globally diagnosable diagnosis submodel design problem is defined as determining, for each $\mathcal{S}_i$, the minimal set of residuals in the complete set of possible residuals, R, to achieve global diagnosability. Formally, the problem is defined as follows:

Problem (Globally Diagnosable Diagnosis Submodel Design)**.** Given n diagnosis submodels, where $\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i})$, $i = \{1, 2, \ldots, n\}$, such that: (*i*) $F = F_1 \cup F_2 \cup \ldots \cup F_n$; and (*ii*) $\forall i \neq j \in \{1, 2, \ldots, n\}$, $F_i \cap F_j = \emptyset$; construct, for each diagnosis submodel, a residual set ${R_i}^+ \subseteq R$ such that: (*iii*) $R_i^+ - R_i$ is minimal; (*iv*) $Y_i^+ \subseteq Y$ are the measurements involved in R_i^+; and (*v*) $\mathcal{S}'_i = (F_i, Y_i^+, R_i^+, L_{F_i,R_i^+})$ is globally diagnosable.

Initial distribution of measurements and faults within the diagnosis submodels are provided by the user. That initial decomposition must fulfill constraints (*i*) and (*ii*) from the design problem, i.e., all faults in the diagnosis model are covered by the diagnosis submodels, and each fault will be included in only one diagnosis submodel. The rest of the constraints must be satisfied by the design approach. The algorithm we present in Section 6 solves the diagnosis submodel design problem and determines the residuals needed for each diagnosis submodel $\mathcal{S}_i$ to become globally diagnosable. Our proposal in this work is to compute such residuals from submodels of the global system model, thus allowing the implementation of residuals in a distributed way. The next section presents our structural model decomposition approach to compute such submodels. Then, in Section 6 we use the submodels to design globally diagnosable diagnosis submodels. Later, in Section 7, we use these globally diagnosable diagnosis submodels to create local diagnosers which are used to define a comprehensive distributed diagnosis architecture.

4. Structural Model Decomposition

As we previously mentioned, residuals can be computed using either the global model of a system or a submodel of this global model with enough analytical redundancy. Structural model decomposition methods, like PCs, have been proposed to decompose a system model into minimal over-determined submodels that are sufficient for fault diagnosis [11, 12, 14]. This decomposition into minimal submodels provides a natural way to distribute the residual generation process into independent computational units, since residuals based on PCs are independent from one another given the measurement set, Y. This approach is fundamentally different from a distributed observer scheme based on a global model [38], where each local observer requires inputs from other local observers.

In this section, we first describe the fundamentals of the PCs approach, and then we generalize PCs to submodels with multiple outputs. Later, in Section 6, we will show how this generalization is necessary for distributed diagnoser design.

[11] In distributed diagnosis, the user typically provides an initial distribution of measurement and fault sets for the desired distribution of diagnosers. This is based on different criteria, e.g, the physical location of the components of the system. However, the approach presented in this paper can also work without this initial partitioning by providing the initial sets of measurements/faults from Possible Conflicts, as explain in the next section.

4.1. Possible Conflicts

In this work, we are interested in generating submodels that allow for the computation of a given set of variables that can be used for residual computation. To do it, the PCs approach considers sensor measurements in the global system as potential local inputs to the submodel. Given the set of potential local inputs (selected from $U \cup Y$) and the set of variables to be computed by the submodel (selected from Y, since the submodels have to estimate measured variables to compute residuals for such variables), we create from a causal model $\mathcal{M}$ a causal submodel $\mathcal{M}_{Y_i}$, in which $Y_i \subseteq Y$ is computed using $C_i \subseteq C$. In this way, each submodel computes its variable values independently from all other submodels. A causal submodel can be defined as follows.

Definition 21 (Causal Submodel)**.** A *causal submodel* $\mathcal{M}_{Y_i}$ of a causal model $\mathcal{M} = (V, C, \mathscr{A})$ is a tuple $\mathcal{M}_{Y_i} = (V_i, C_i, \mathscr{A}_i)$, where $V_i \subseteq V$, $C_i \subseteq C$, and $\mathscr{A}_i \cap \mathscr{A} \neq \varnothing$.

When using measurements (from Y) as local inputs for a causal submodel, the causality of these constraints must be reversed, and so, in general, $\mathscr{A}_i$ is not a subset of $\mathscr{A}$.[12] All remaining causal assignments in $\mathscr{A}_i$ will still be found in $\mathscr{A}$.

A PC is a special kind of causal submodel, where the output set, $Y_i \subseteq Y$, is a singleton, and the set of constraints C_i is the minimal set of constraints necessary to compute Y_i. Hence, a PC is minimal regarding its constituent constraints. This way, PCs are capable of computing, off-line, the set of minimal conflicts in a system model. Formally, a PC can be defined as follows:

Definition 22 (Possible Conflict)**.** A causal submodel, $\mathcal{M}_{Y_i} = (V_i, C_i, \mathscr{A}_i)$, is a *possible conflict* if $|Y_i| = 1$, and there is no other causal submodel, $\mathcal{M}_{Y_j} = (V_j, C_j, \mathscr{A}_j)$, such that $C_j \subseteq C_i$.

Example 15. Consider the three-tank system model. A PC, $\mathcal{M}_{h_1}$, can be computed for the level in tank 1, h_1, since h_1 is a measured variable (Figure 5a shows the causal graph of $\mathcal{M}_{h_1}$). Variable h_1 can be computed from causal constraint α_4. Constraint α_4 needs the value of p_1, which is computed from the causal constraint α_1. Since u_1 is an input variable, which is known, and p_1 is already computed by α_1, the only unknown variable within α_1 is p_2. In the global model approach, p_2 is computed from causal constraint α_2. However, in the PCs approach, since the measurements are considered as inputs to the system, h_2 is an input variable, and p_2 can by computed from h_2 by inverting causality in α_5. No unknown variables remain to be computed, consequently we have the PC $\mathcal{M}_{h_1}$. This PC can then be used to compute a residual for the level of tank 1, with a reduced number of constraints than the global system model.

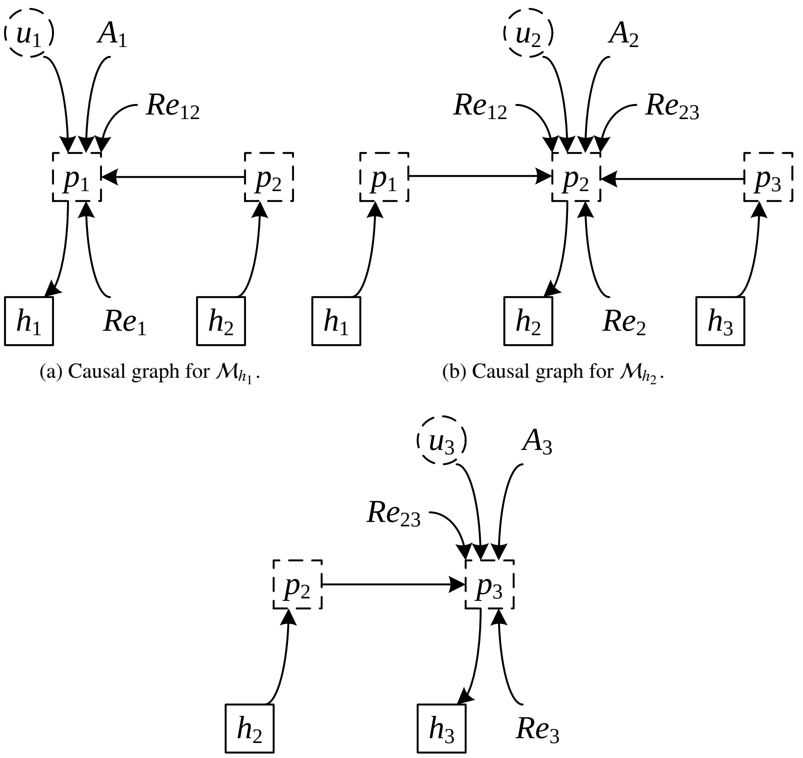
The procedure for generating a PC from a causal model is given as Algorithm 1 [18]. Given a causal model $\mathcal{M}$, and an output variable to be computed y, the `GeneratePC` algorithm derives a causal submodel $\mathcal{M}_i$ that computes y using as local inputs only variables from $U^* = U \cup (Y - \{y\})$. We provide here a simplified version of the algorithm presented in [18], and refer the reader to [18] for the extended algorithm and additional details. We briefly summarize the algorithm below.

In Algorithm 1, the *variables* queue represents the set of variables that have been added to the submodel but have not yet been resolved, i.e., they cannot yet be computed by the submodel. This queue is initialized to $\{y\}$, and the algorithm then iterates until this queue has been emptied, i.e., the submodel can compute y using only variables in U^*. For each variable v that must be resolved, we use the `GetBestConstraint` subroutine (Subroutine 2) to find the constraint that should be used to resolve v in the minimal way.

The `GetBestConstraint` subroutine tries to find a constraint that *completely* resolves the variable, i.e., resolves v without further backward propagation (all other variables involved in the constraint are in $V_i \cup \Theta \cup U^*$). Such a constraint may be the one that computes v in the current causality, if all needed variables are already in the submodel (in V_i) or are available local inputs (in U^*); or such a constraint may be one that computes a measured output $y^* \in U^*$, in which case the causality will be modified such that y^* becomes an input, i.e., the constraint in the new causality will compute v rather than y^*. If no such constraint exists, then the constraint that computes v in the current causal assignment is chosen, and further backward propagation will be necessary.

Clearly, there are many submodels that compute any given y using a given U^*. The global model is one of these solutions. Algorithm 1 finds a *minimal* (with respect to subset of constraints) submodel that satisfies this, which is

[12] This process of causality inversion in the measurements is sometimes known as sensor dualization [39].



(a) Causal graph for $\mathcal{M}_{h_1}$.

(b) Causal graph for $\mathcal{M}_{h_2}$.

(c) Causal graph for $\mathcal{M}_{h_3}$.

Figure 5: Possible conflicts of the three-tank system.

guaranteed in Subroutine 2 by resolving a variable without further backward propagation whenever possible. Since submodels computed by the algorithm are minimal, they are by definition possible conflicts. There may be multiple submodels that are equally minimal (i.e., due to a choice of which local input to use), and the algorithm returns the first that it finds.

The algorithm also generates only *complete* submodels, i.e., the submodels contain at least the variables needed to compute its y. This is guaranteed because the algorithm only stops propagation at variables included in $V_i \cup \Theta \cup U^*$ [18].

In the worst case, the algorithm must visit all variables and constraints. On each variable, Subroutine 2 is called, which in the worst case considers all variables in $Y \cup U^*$. So the overall worst-case time complexity is $O((|V| + |E|) \cdot |Y \cap U^*|)$. Since $(Y \cap U^*) \subset V$, the algorithm is polynomial in the model size. On average the connectivity among constraints and variables will not be high, and the complexity will be much lower in practice.

Example 16. Applying algorithm 1,[13] to the three-tank system model, we find a set of three minimal submodels, each one of them estimating the level in one of the tanks (Figure 5 shows the causal graphs of these three PCs):

$$\begin{aligned}
\mathcal{M}_{h_1} &= (\{\{p_1\} \cup \{A_1, Re_1, Re_{12}\} \cup \{u_1, h_2\} \cup \{h_1\}\}, \{c_1, c_4, c_5\}, \{\alpha_1, \alpha_4, \alpha_5\}) \\
\mathcal{M}_{h_2} &= (\{\{p_2\} \cup \{A_2, Re_2, Re_{12}, Re_{23}\} \cup \{u_2, h_1, h_3\} \cup \{h_2\}\}, \{c_2, c_4, c_5, c_6\}, \{\alpha_2, \alpha_4, \alpha_5, \alpha_6\}) \\
\mathcal{M}_{h_3} &= (\{\{p_3\} \cup \{A_3, Re_3, Re_{23}\} \cup \{u_3, h_2\} \cup \{h_3\}\}, \{c_3, c_5, c_6\}, \{\alpha_3, \alpha_5, \alpha_6\}).
\end{aligned}$$

The main advantage of the PC submodels is that, since they estimate a measured variable, they will be the source of conflicts, i.e., they can be used to compute a residual for that variable with a reduced number of constraints compared

[13] Another algorithm for deriving PCs is given in [14]. This algorithm computes submodels considering all possible causal assignments, what makes it more inefficient than the algorithm presented in this paper.

Algorithm 1 $\mathcal{M}_i$ = GeneratePC($\mathcal{M}, y$)

```
$U^* \leftarrow U \cup (Y - \{y\})$
$V_i \leftarrow \{y\}$
$C_i \leftarrow \varnothing$
$\mathcal{A}_i \leftarrow \varnothing$
$variables \leftarrow V_i$
while $variables \neq \varnothing$ do
    $v \leftarrow$ pop($variables$)
    $c \leftarrow$ GetBestConstraint($v, V_i, U^*, \mathcal{A}$)
    $C_i \leftarrow C_i \cup \{c\}$
    $\mathcal{A}_i \leftarrow \mathcal{A}_i \cup \{(c, v)\}$
    for all $v' \in V_c$ do
        if $v' \notin V_i$ and $v' \notin \Theta$ and $v' \notin U^*$ then
            $variables \leftarrow variables \cup \{v'\}$
        end if
        $V_i \leftarrow V_i \cup \{v'\}$
    end for
end while
$\mathcal{M}_i \leftarrow (V_i, C_i, \mathcal{A}_i)$
```

Subroutine 2 c = GetBestConstraint($v, V_i, U^*, \mathcal{A}$)

```
$c_v \leftarrow$ find $c$ where $(c, v) \in \mathcal{A}$
if $V_{c_v} - v \subseteq V_i \cup U^*$ then
    return $c_v$
else
    for all $y^* \in Y^* \cap U^*$ do
        $c_{y^*} \leftarrow$ find $c$ where $(c, y^*) \in \mathcal{A}$
        if $v \in V_{c_{y^*}}$ and $V_{c_{y^*}} - v \subseteq V_i \cup U^*$ then
            return $c_{y^*}$
        end if
    end for
end if
return $c_v$
```

to the global system model. We denote a residual computed from a PC as $r_{y(Y_i)}$, where y is the measurement estimated by the residual, and Y_i refers to the submodel with those measurements as outputs, $\mathcal{M}_{Y_i}$. For example, $r_{h_1(h_1)}$ denotes the residual that estimates h_1 from submodel $\mathcal{M}_{h_1}$.

Similarly to what is shown in Section 2, fault signatures can be computed for the residuals of the PCs. However, there is an important difference. Since residual orderings may be defined only within a given submodel, and residuals are now decoupled in the submodels, we cannot define the orderings between residuals in different submodels. For example, we cannot derive an ordering between $r_{h_1(h_1)}$ and $r_{h_2(h_2)}$ for Re_{12}^+.

Example 17. The fault signatures derived for the residual set $R = \{r_{h_1(h_1)}, r_{h_2(h_2)}, r_{h_3(h_3)}\}$. computed from the PCs, are shown in Table 2. In this case, the PCs are able to decouple the effect of faults in the residuals, and so each residual is only affected by a subset of the faults. For example, a decrease in the capacitance of tank 1, denoted by A_1^-, causes a discontinuous increase in the residuals related to tank 1 pressure, $r_{h_1(h_1)}$, followed by a smooth decrease, denoted by the signature $+-$. But fault A_1^- has no effect on residuals $r_{h_2(h_2)}$ and $r_{h_3(h_3)}$, as denoted by the 00 signature, since the constraint c_1, which contains fault parameter A_1, is not included in $\mathcal{M}_{h_2}$ or $\mathcal{M}_{h_3}$.

Consider the residual set given for the global model, $\mathcal{M}$, of the three-tank system (Table 1). A diagnosis model defined with these residuals is diagnosable. However, given the residual set for the PCs (see Table 2), the diagnosis model is not diagnosable since fault Re_1^+ cannot be distinguished from fault Re_{12}^+, and fault Re_2^+ cannot be distinguished from fault Re_{23}^+. For example, if Re_1^+ occurs, then a 0+ will be observed on $r_{h_1(h_1)}$ activation. At this point, that observation is consistent with both Re_1^+ and Re_{12}^+ occurring, and a diagnoser would not able to determine which has

Table 2: Fault signatures and relative residual orderings for the set of PCs, $\mathcal{M}_{h_1}$, $\mathcal{M}_{h_2}$, and $\mathcal{M}_{h_3}$, of the three-tank system.

Fault	$r_{h_1(h_1)}$	$r_{h_2(h_2)}$	$r_{h_3(h_3)}$	Residual Orderings
A_1^-	$+-$	00	00	$\varnothing$
Re_1^+	$0+$	00	00	$\varnothing$
Re_{12}^+	$0+$	$0-$	00	$\varnothing$
A_2^-	00	$+-$	00	$\varnothing$
Re_2^+	00	$0+$	00	$\varnothing$
Re_{23}^+	00	$0+$	$0-$	$\varnothing$
A_3^-	00	00	$+-$	$\varnothing$
Re_3^+	00	00	$0+$	$\varnothing$

occurred, because the deviation in $r_{h_2(h_2)}$ has not manifested yet. Since we would have to wait infinitely long to ensure that no deviation occurs, Re_1^+ cannot be distinguished from Re_{12}^+. Consequently, with PCs we can decompose the diagnosis model into smaller submodels, which can be used to implement residuals in a distributed way. However, this decomposition can decrease the diagnosability compared to the diagnosis model. To avoid this problem, our proposal in this work is to merge PCs, when necessary, so that distributed diagnosers can be implemented in a distributed way, maintaining the diagnosability properties of the global diagnosis model, and without a central coordinator. Next, we show our proposal for submodel merging. Later, in Section 6, we show how this merging process can be used to design globally diagnosable diagnosis submodels.

4.2. Multi-output Possible Conflicts

By definition, possible conflicts are submodels computed as minimal subsets of constraints that estimate a single measurement. Therefore, one PC is derived for each system measurement, and consequently can only be used to compute one residual. In this work, we propose to use the PCs to derive submodels computed as minimal subsets of constraints that estimate several measurements. We call these submodels *multi-output PCs*.

Definition 23 (Multi-output PCs)**.** A causal submodel, $\mathcal{M}_{Y_i} = (V_i, C_i, \mathcal{A}_i)$, is a *multi-output PC* if $|Y_i| > 1$, and there is no other causal submodel, $\mathcal{M}_{Y_j} = (V_j, C_j, \mathcal{A}_j)$, such that $Y_i = Y_j$ and $C_j \subseteq C_i$.

The advantage of generating multi-output PCs is that additional residuals may then be defined for all the measurements within these multi-output PCs (not just one measurement as in the PCs), but at the same time will have a smaller size than the global model. Multi-output PCs can be derived by merging two or more PCs. The merge operation $\oplus$ between two causal submodels is defined by Algorithm 3. As shown in the algorithm, the merged submodel must have all the states, outputs, parameters, and constraints of its constituent submodels, and must have all the inputs, minus those that have become outputs in the merged submodel. For the causal assignments, the merged submodel retains all of the causal assignments of its constituent submodels except for those that are different in the two submodels. This can happen only due to causality inversion of outputs in the global model, causing them to become inputs in the submodel. The causal assignment for the related constraints must be in the form where the output variable belongs to the output set of the merged submodel. Recall that we denote merged submodels by the set of outputs they compute, e.g., the submodel formed by merging PCs $\mathcal{M}_{h_1}$ and $\mathcal{M}_{h_2}$ is denoted as $\mathcal{M}_{h_1,h_2}$.[14]

Example 18. Figure 6 shows the causal graph of the multi-output PC $\mathcal{M}_{h_1,h_2}$. This submodel estimates two output variables h_1 and h_2, contains two state variables p_1 and p_2, and the only measurement used as input is h_3. A residual may be defined for each measurement in each submodel. Recall that we denote a residual as $r_{y(Y_i)}$, where y is the measurement estimated by the residual, and Y_i refers to the submodel with those measurements as outputs. Hence, $r_{h_1(h_1,h_2)}$

[14] Each residual is computed using a minimal submodel, i.e. a PC with only one output. When we merge several PCs, we have several outputs, but the PC is still minimal in the sense that there is no subset of constraints capable to estimate the same set of outputs. As a consequence, given the merging of two submodels $\mathcal{M}_{h_1}$ and $\mathcal{M}_{h_2}$, we can make reference to the resulting submodel as $\mathcal{M}_{h_1,h_2}$ without confusion (due to the kind of causal model we are using).

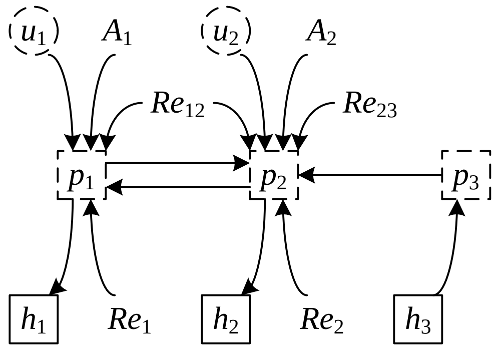


Figure 6: Multi-output PC $\mathcal{M}_{h_1,h_2}$ for the three-tank system.

denotes the residual that estimates the variable h_1 from submodel $\mathcal{M}_{h_1,h_2}$, and $r_{h_2(h_1,h_2)}$ the residual that estimates the variable h_2 from submodel $\mathcal{M}_{h_1,h_2}$

Example 19. Using Algorithm 3 for the three-tank system, we obtained the following multi-output PCs:

$$\mathcal{M}_{h_1,h_2} = (\{\{p_1, p_2\} \cup \{A_1, A_2, Re_1, Re_2, Re_{12}, Re_{23}\} \cup \{u_1, u_2, h_3\} \cup \{h_1, h_2\}\}, \{c_1, c_2, c_4, c_5, c_6\}, \{\alpha_1, \alpha_2, \alpha_4, \alpha_5, \alpha_6\})$$
$$\mathcal{M}_{h_1,h_3} = (\{\{p_1, p_3\} \cup \{A_1, A_3, Re_1, Re_3, Re_{12}, Re_{23}\} \cup \{u_1, u_3, h_2\} \cup \{h_1, h_3\}\}, \{c_1, c_3, c_4, c_5, c_6\}, \{\alpha_1, \alpha_3, \alpha_4, \alpha_5, \alpha_6\})$$
$$\mathcal{M}_{h_2,h_3} = (\{\{p_2, p_3\} \cup \{A_2, A_3, Re_2, Re_3, Re_{12}, Re_{23}\} \cup \{u_3, u_3, h_1\} \cup \{h_2, h_3\}\}, \{c_2, c_3, c_4, c_5, c_6\}, \{\alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6\}).$$

As an example, Table 3 shows the fault signatures and residual orderings derived for submodels $\mathcal{M}_{h_1,h_2}$ and $\mathcal{M}_{h_3}$, i.e., PCs $\mathcal{M}_{h_1}$ and $\mathcal{M}_{h_2}$ are merged. For this scenario, $R = \{r_{h_1(h_1,h_2)}, r_{h_2(h_1,h_2)}, r_{h_3(h_3)}\}$ and now we have a trade-off between using the global model and using minimal submodels. As a consequence of merging $\mathcal{M}_{h_1}$ and $\mathcal{M}_{h_2}$, residual orderings can now be computed between residuals $r_{h_1(h_1,h_2)}$ and $r_{h_2(h_1,h_2)}$.

Table 3: Fault signatures and relative residual orderings for the submodels $\mathcal{M}_{h_1,h_2}$ and $\mathcal{M}_{h_3}$, of the three-tank system.

Fault	$r_{h_1(h_1,h_2)}$	$r_{h_2(h_1,h_2)}$	$r_{h_3(h_3)}$	Residual Orderings
A_1^-	$+-$	$0+$	$0\,0$	$r_{h_1(h_1,h_2)} \prec r_{h_2(h_1,h_2)}$
Re_1^+	$0+$	$0+$	$0\,0$	$r_{h_1(h_1,h_2)} \prec r_{h_2(h_1,h_2)}$
Re_{12}^+	$0+$	$0-$	$0\,0$	$\varnothing$
A_2^-	$0+$	$+-$	$0\,0$	$r_{h_2(h_1,h_2)} \prec r_{h_1(h_1,h_2)}$
Re_2^+	$0+$	$0+$	$0\,0$	$r_{h_2(h_1,h_2)} \prec r_{h_1(h_1,h_2)}$
Re_{23}^+	$0+$	$0+$	$0-$	$r_{h_2(h_1,h_2)} \prec r_{h_1(h_1,h_2)}$
A_3^-	$0\,0$	$0\,0$	$+-$	$\varnothing$
Re_3^+	$0\,0$	$0\,0$	$0+$	$\varnothing$

Now that we have presented the fundamental ideas of structural model decomposition, PCs, and our proposal for PC merging into multi-output PCs, the next section presents our distributed diagnosis architecture using PCs.

5. Distributed Diagnosis Architecture

The architecture of the proposed distributed diagnosis framework is shown in Fig. 7. This framework has been implemented in MATLAB and it runs, given the inputs, in a completely automatic way. In the architecture, each local diagnoser takes a subset of the inputs, U_i, and outputs, Y_i, of the system and provides diagnosis results, i.e., U_1 and Y_1 represent a subset of the inputs and a subset of the outputs, respectively, used by local diagnoser 1. For each one of the

Algorithm 3 $\mathcal{M}_{Y_{i,j}} = \mathcal{M}_{Y_i} \oplus \mathcal{M}_{Y_j}$

```
$X_{i,j} \leftarrow X_i \cup X_j$
$\Theta_{i,j} \leftarrow \Theta_i \cup \Theta_j$
$U_{i,j} \leftarrow (U_i \cup U_j) - (Y_i \cup Y_j)$
$Y_{i,j} \leftarrow Y_i \cup Y_j$
$V_{i,j} \leftarrow X_{i,j} \cup \Theta_{i,j} \cup U_{i,ju} \cup Y_{i,j}$
$C_{i,j} \leftarrow C_i \cup C_j$
$\mathcal{A}_{i,j} \leftarrow \varnothing$
for $c \in C_{i,j}$ do
    if $c \in C_i$ and $c \notin C_j$ then
        $\alpha_{i,j} \leftarrow$ find $(c, v) \in \mathcal{A}_i$
    else if $c \notin C_i$ and $c \in C_j$ then
        $\alpha_{i,j} \leftarrow$ find $(c, v) \in \mathcal{A}_j$
    else
        $v_i \leftarrow$ find $v$ where $(c, v) \in \mathcal{A}_i$
        $v_j \leftarrow$ find $v$ where $(c, v) \in \mathcal{A}_j$
        if $v_i \in Y_{i,j}$ then
            $\alpha_{i,j} \leftarrow (c, v_i)$
        else if $v_j \in Y_{i,j}$ then
            $\alpha_{i,j} \leftarrow (c, v_j)$
        else
            $\alpha_{i,j} \leftarrow (c, v_i)$
        end if
    end if
    $\mathcal{A}_{i,j} \leftarrow \mathcal{A}_{i,j} \cup \{\alpha_{i,j}\}$
end for
$M_{Y_{i,j}} = (V_{i,j}, C_{i,j})$
```

local diagnosers we can distinguish between the off-line modules (in dashed lines) and the on-line modules (in solid lines). The off-line modules are related with the design of the distributed diagnosis system, and are used to implement the on-line modules that are used for the diagnosis process.

Regarding the off-line modules, we start with a global model of the system, and compute the Possible Conflicts. The PCs are computed using the approach presented in Section 4, however, our framework is independent of this, and any other structural decomposition method could be used to decompose the system instead of the PCs [29]. Based on the computed PCs and the initial fault/measurements partition provided by the user, the *diagnosis submodel design* module (Section 6), first computes the initial submodels for each diagnosis submodel (by merging PCs if necessary), and then determines the PCs that need to be merged with the initial submodels to create globally diagnosable diagnosis submodels. Once the globally diagnosable diagnosis submodels have been designed, the result is used in three different modules: (*i*) the *residual design* module (Section 6), which implements the residual generators for system tracking for each local diagnoser; (*ii*) the *diagnoser implementation* module (Section 7), which implements the event-based local diagnosers for fault isolation; and (*iii*) the *local parameter estimator design* module (Section 8), which implements the local parameter estimators for each fault within each local diagnoser for fault identification. There is one main assumption in our approach, that is also present in other distributed diagnosis approaches, such as [40]:

Assumption 4. The observation system is complete, reliable, and all symbol generation is correct.

As a consequence we assume we can observe every event, we do not miss observations, and the communication process is efficient. Additionally, we can also state another assumption:

Assumption 5. A communication channel between two components is bounded.

Hence the communication will not introduce new or spurious states in the system.

Regarding the on-line modules, our implementation is similar to the qualitative fault isolation (QFI) framework described in [7, 9], but in this case we use submodels for fault detection and identification, instead of the global

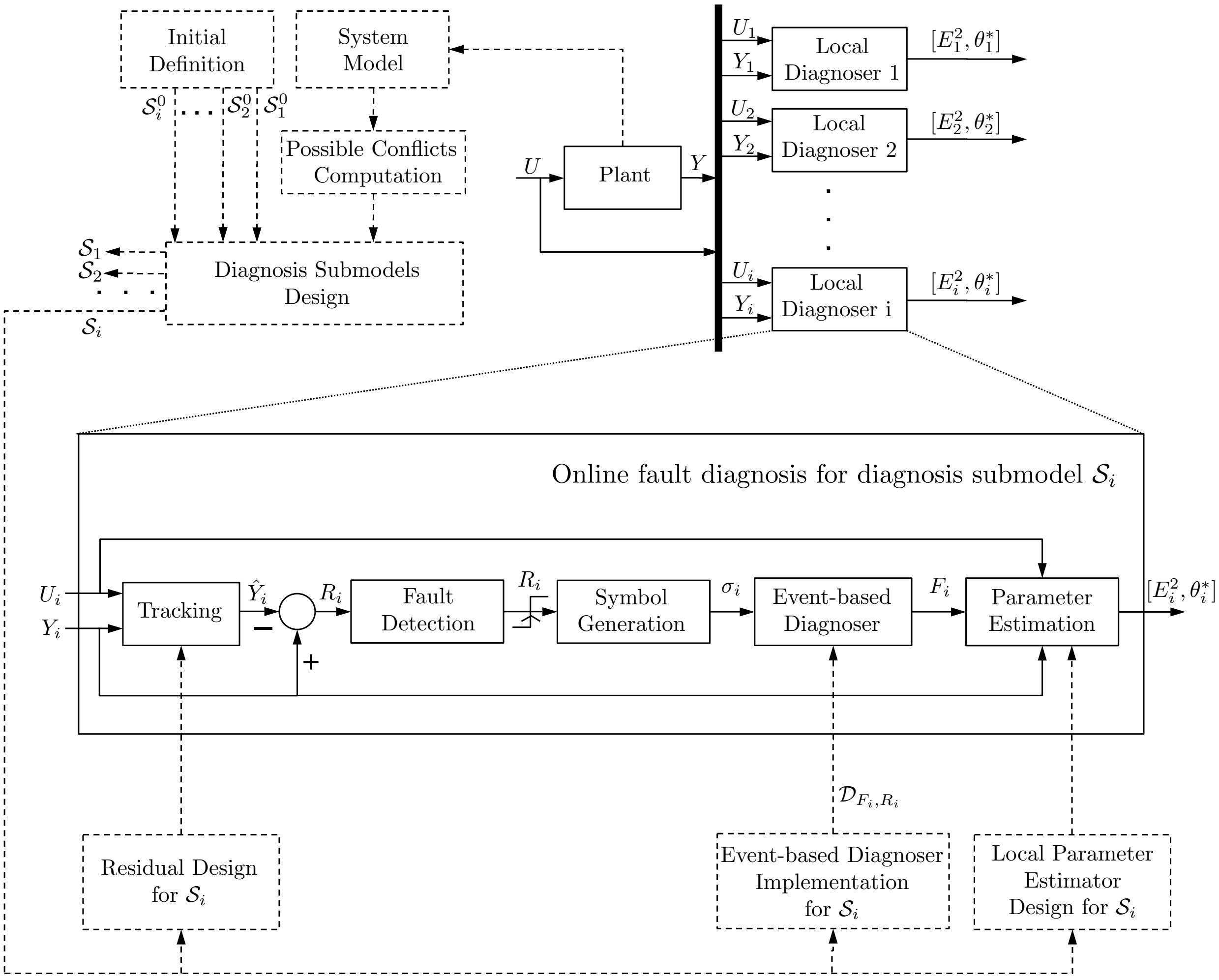


Figure 7: Architecture of the event-based distributed diagnosis approach.

model. First, the *tracking* module estimates the output measurements, $\hat{Y}_i$, defined for each local diagnoser. Tracking is performed in discrete-time using standard techniques. Typically, a robust method, such as the extended or unscented Kalman filter [41], or the particle filter [42], provides tracking of the nominal system behavior in the presence of process noise and discretization error. Then, output estimates $\hat{Y}_i$ are compared against the real measurements of the system Y_i using a statistical test within the *fault detection* module. In our approach, the *fault detection* module employs the Z-test to look for nonzero residual signals, as described in [43]. Because we use a simulation for predictions, fault detection can be achived very quickly. A statistically significant nonzero value for the residual, R_i, indicates a fault occurrence and triggers the *symbol generation* module, where the measurement and slope values of the residual signals, R_i, are converted to qualitative values, σ_i. These deviations are then used by the *event-based diagnoser* to isolate faults. Fault candidates whose predicted signatures remain consistent with the observed measurements are considered fault candidates, F_i, others are dropped. Finally, from the subset F_i of fault candidates, the *parameter estimation* block performs fault identification to determine the fault that has occurred in the system (described by the parameter associated to the fault, θ_i^*), together with its estimation accuracy, E_i^2. In our framework, optimization is done by using the local parameter estimator and iteratively modifying the value of the parameter to minimize the difference between the estimated outputs and the measured outputs, although any other appropriate parameter estimation technique could be used. The next three sections develop further each part of the proposed architecture.

6. Globally Diagnosable Diagnosis Submodel Design

Recall that the globally diagnosable diagnosis submodel design problem is to find for each diagnosis submodel the minimal set of residuals needed to make the diagnosis submodel globally diagnosable. A solution to this problem is one in which each diagnosis submodel is globally diagnosable. An optimal solution is one in which the number of residuals assigned to each diagnosis submodel is minimal, i.e., there is no smaller set of residuals for any one diagnosis submodel for which it is also globally diagnosable. There may be several optimal solutions.

We can treat this design problem as a search problem, where the search space for a given diagnosis submodel is defined by the residuals that may be added to the diagnosis submodel. However, this space is extremely large, because the number of possible submodels, and, consequently, the size of the complete residual set, grows exponentially with the number of measurements. For a system with $|Y|$ measurements the number of possible submodels is $2^{|Y|} - 1$, and the number of residuals over all possible submodels is $|Y| \times 2^{|Y|-1}$.

The advantage of PCs is that, since measurements are used as local inputs to decompose the system, they are capable of decoupling the effects of faults in the system. This results in an improvement in diagnosability in a local sense. The intuition, then, is that including PC-based residuals will lead to improved diagnosis submodel design (i.e., fewer residuals) because of this improvement in diagnosability, thus requiring less search overall.

Example 20. Let us assume that the three-tank system is split into three diagnosis submodels, $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, where for $\mathcal{S}_1$, $F_1 = \{A_1^-, Re_1^+, Re_{12}^+\}$, $Y_1 = \{h_1\}$; for $\mathcal{S}_2$, $F_2 = \{A_2^-, Re_2^+, Re_{23}^+\}$, $Y_2 = \{h_2\}$; and for $\mathcal{S}_3$, $F_3 = \{A_3^-, Re_3^+\}$, $Y_3 = \{h_3\}$. Also assume that we use the $\mathcal{M}$-based residuals, so $R_1 = \{r_{h_1}\}$, $R_2 = \{r_{h_2}\}$, and $R_3 = \{r_{h_3}\}$. Analyzing global diagnosability, we see that none of the diagnosis submodels is globally diagnosable, i.e., we will have to add new residuals to each diagnosis submodel in order to satisfy our design constraints. For example, consider diagnosis submodel $\mathcal{S}_3$. Looking at Table 1, we see that the diagnosis submodel is not globally diagnosable, since effects produced by Re_3^+ on r_{h_3} are not unique. Now assume we use the PC-based residuals, $R_1 = \{r_{h_1(h_1)}\}$, $R_2 = \{r_{h_2(h_2)}\}$, and $R_3 = \{r_{h_3(h_3)}\}$. We see that now $\mathcal{S}_3$ is globally diagnosable because only one nonlocal fault, Re_{23}^+, produces an effect on $r_{h_3(h_3)}$, and it is a different effect from those produced by the local faults. So if $\mathcal{S}_3$ uses the PC-based residual instead of the global model-based residual, it produces an improved diagnosis submodel design. However, the other diagnosis submodels are still not globally diagnosable, and cannot be made so by including any other PC-based residual, because those diagnosis submodels contain the faults that make the system as a whole nondiagnosable using only the PC-based residuals.

As shown in the previous example, PC-based residuals can improve global diagnosability over the global model in some diagnosis submodels, but typically it is not enough to guarantee global diagnosability for all diagnosis submodels. This suggests that we require a more general approach that considers residuals from the complete set, R, considering all possible submodels.

As previously pointed out, the size of the search space is dependent on the number of possible residuals, and the complete set is very large. Further, there is much overlap of information between the different residuals, for example, compare Tables 1, 2, and 3. So, instead of searching over the residual space ($|Y| \times 2^{|Y|-1}$), we perform a search over the measurement space ($2^{|Y|} - 1$), which is much smaller than the residual space, and define residuals in a particular way. Specifically, given a set of measurements Y_i, we use the residuals for the submodel that includes exactly the measurements in Y_i as outputs, i.e., for Y_i we use the residual set $\{r_{y(Y_i)} : y \in Y_i\}$ (recall that y refers to the measured variable estimated by the residual, and Y_i refers to the submodel with those measurements as outputs, $\mathcal{M}_{Y_i}$, used to compute the residual). For example, if $Y_i = \{h_1, h_2\}$, then $R_i = \{r_{h_1(h_1,h_2)}, r_{h_2(h_1,h_2)}\}$. This approach significantly reduces the search space, because we accept only diagnosis submodel definitions for which the residuals are those from the submodel defined by the diagnosis submodel measurements. Further, by defining residuals in this way, each local diagnoser will have its own submodel to compute residuals independently of other local diagnosers. This is in contrast to an approach where residuals are computed in a centralized way [8, 10], and allows the local diagnosers to be independent of each other for residual generation. As previously described in Section 3, the diagnosers will become independent of each other with respect to fault isolation once they have been designed to be globally diagnosable.

The search problem for a diagnosis submodel is now defined by (*i*) the initial state, which is from the user-provided initial diagnosis submodel definition;[15] (*ii*) the successor function, which adds a measurement to the diagnosis submodel, thus reformulating the submodel and defining a new set of residuals, (*iii*) the goal state, which is a derived

[15] The initial diagnosis submodel definition is provided by the user based on a desired distribution of the components in the system. Typically,

diagnosis submodel that is globally diagnosable; and (*iv*) the path cost, which is the number of measurements associated with the diagnosis submodel. The search problem is to find the globally diagnosable extension of the given diagnosis submodel with the fewest measurements added to all of the diagnosis submodels.

Using this formulation, breadth-first search (BFS) will find the optimal solution since the path costs are uniform. BFS will also find the optimal solution using the alternate formulation that searches over the complete residual space. As a much more efficient alternative, we present here a greedy algorithm that, in our experience, has a high probability of also finding the optimal solution. Our heuristic is to compute the number of faults which are not globally distinguishable when adding a proposed measurement. The locally optimal action is to add the measurement that minimizes this number.

The greedy diagnosis submodel design approach is shown in Algorithm 4. The algorithm is provided as inputs an initial assignment of faults and measurements to diagnosis submodels. It first computes the single-output PCs for all the measurements, storing in $\mathbb{M}$. For each diagnosis submodel, we first construct the multi-output PC for its current set of measurements using the submodel merging algorithm, and then extract its residuals (recall that $r_{y(Y_i)}$ refers to the residual for measurement y in submodel $\mathcal{M}_{Y_i}$). We then compute the subset of measurements over which we will consider adding to the diagnosis submodel as $Y' = Y - Y_i$.[16] We then identify the best measurement within Y' to add to Y_i, using the `FindBestY` subroutine, which implements our heuristic. For each possible measurement to add, this subroutine constructs the new set of residuals, then determines the faults, F_i^*, that are not globally distinguishable for the diagnosis submodel and this residual set. To do this, the algorithm constructs new diagnosis submodels by merging the previous submodel, $\mathcal{M}_{Y_i}$, with the PC for the new measurement, $\mathcal{M}_y$. The fault language for the new diagnosis submodel is then derived to establish distinguishability properties. The measurement that results in the smallest F_i^* (this is the *score* variable in our algorithm) is selected as the best measurement and becomes the output y_i^*. We then update Y_i, reconstruct the residual set for the new measurement set, and continue in this fashion until $\mathcal{S}_i$ is globally diagnosable.

`GenerateLocalLanguage`, given as Subroutine 6, performs a syntactic transformation on the global fault language for a model to derive a local fault language for its submodel. Within each trace, the fault signature is replaced by the version for the submodel by changing the subscript to that of the submodel. This is only a syntactic transformation; here we take advantage of the fact that residuals are defined in a consistent way between models and submodels, where the residual deviation for a measurement in a local submodel will be the same as its deviation in the global model (e.g., compare Tables 1 and 3), and if the measurement does not appear in the local submodel, there will be no associated residual deviation (it is replaced by the empty trace ϵ).

Based on our assumptions, the design algorithm will always find globally diagnosable diagnosis submodels, as stated in Theorem 1. This result applies equally to the BFS version of the design algorithm.

Theorem 1. *For a set of n diagnosis submodels $\{\mathcal{S}_i = (F_i, Y_i, R_i, \varnothing) : i = 1, \ldots, n\}$, Algorithm 4 always returns globally diagnosable diagnosis submodels.*

Proof. For a given diagnosis submodel, $\mathcal{S}_i$, the algorithm continually adds $y \in Y$ to Y_i until the diagnosis submodel is globally diagnosable. This will always terminate because, in the worst case, all $y \in Y$ are added to Y_i, in which case the global model is recovered, which is diagnosable (Assumption 3 in Section 3). □

The worst-case performance of both algorithms occurs when all the system measurements are added to a diagnosis submodel, which refers to the case where the diagnosis problem cannot be decomposed. In the worst case, the BFS algorithm explores a space of size $O(2^{|Y|})$, whereas the greedy algorithm explores only a space of size $O(|Y|)$. With appropriate sensors, this is a very unlikely scenario. Because the greedy algorithm adds measurements incrementally, it is, in general, non-optimal, but here we trade off optimality for computational efficiency. However, as we will see in the following examples and in the case study, the greedy algorithm still obtains the optimal solution as determined by

this distribution is based on the proximity of the components and often the physical connectivity in the system. For example, in the three-tank system, diagnosis submodel $\mathcal{S}_1$ is defined as the tank 1 and its connecting pipes. The PC decomposition algorithms can also help the user to define an initial coupling of components into diagnosis submodels.

[16] If $Y - Y_i$ is too large, a subset of the measurements that are considered to be more useful, according to some heuristic, can be used instead. An example of such a heuristic is the diagnosis submodel distance heuristic developed in [8], in which preference is given to measurements that are in neighboring diagnosis submodels.

Algorithm 4 $\{S_i^*\}_{i=1}^n$ = `GreedyDesign`$(\mathcal{M}, L_{F,R}, \{S_i = (F_i, Y_i, \varnothing, \varnothing)\}_{i=1}^n)$

```
  $\mathbb{M} \leftarrow \varnothing$
  **for all** $y \in Y$ **do**
    $\mathcal{M}_y \leftarrow$ GeneratePC$(\mathcal{M}, y)$
    $\mathbb{M} \leftarrow \mathbb{M} \cup \{\mathcal{M}_y\}$
  **end for**
  **for all** $\mathcal{S}_i \in \mathbb{S}$ **do**
    $\mathcal{M}_{Y_i} \leftarrow (\varnothing, \varnothing)$
    **for all** $y_i \in Y_i$ **do**
      $\mathcal{M}_{Y_i} \leftarrow \mathcal{M}' \oplus \mathcal{M}_{y_i}$
    **end for**
    $R_i \leftarrow \{r_{y(Y_i)} : y \in Y_i\}$
    $L_{F_i,R_i} \leftarrow$ GenerateLocalLanguage$(L_{F,R}, Y, F_i, R_i, Y_i)$
    **while** $\mathcal{S}_i$ not globally diagnosable **do**
      $Y' \leftarrow Y - Y_i$
      $(y_i^*, \mathcal{M}_i^*) \leftarrow$ FindBestY$(\mathbb{M}, L_{F,R}, \mathcal{M}_{Y_i}, F, F_i, Y', Y_i)$
      $Y_i \leftarrow Y_i \cup \{y_i^*\}$
      $\mathcal{M}_{Y_i} \leftarrow \mathcal{M}_i^*$
      $R_i \leftarrow \{r_{y(Y_i)} : y \in Y_i\}$
      $L_{F_i,R_i} \leftarrow$ GenerateLocalLanguage$(L_{F,R}, Y, F_i, R_i, Y_i)$
    **end while**
    $S_i^* \leftarrow (F_i, Y_i, R_i, L_{F_i,R_i})$
  **end for**
```

Subroutine 5 $(y_i^*, \mathcal{M}_i^*) \leftarrow$ `FindBestY`$(\mathbb{M}, L_{F,R}, \mathcal{M}_{Y_i}, F, F_i, Y, Y_i)$

```
  **for all** $y \in Y - Y_i$ **do**
    $M_i^* \leftarrow \mathcal{M}_{Y_i} \oplus \mathcal{M}_y$
    $R_i \leftarrow \{r_{y'(Y_i \cup \{y\})} : y' \in Y_i \cup \{y\}\}$
    $L_{F_i,R_i} \leftarrow$ GenerateLocalLanguage$(L_{F,R}, Y, F_i, R_i, Y_i)$
    $F_i^* \leftarrow \{f_i : f_i \sim_{R_i} f_j$ for $f_i \in F_i, f_j \in F,$ and $f_i \neq f_j\}$
    $score_y \leftarrow |F_i^*|$
  **end for**
  $y_i^* \leftarrow y$ such that $score_y$ is minimum
```

Subroutine 6 $L_{F_i,R_i} \leftarrow$ `GenerateLocalLanguage`$(L_{F,R}, Y, F_i, R_i, Y_i)$

```
  **for all** $f \in F_i$ **do**
    $L_{f,R_i} \leftarrow \varnothing$
    **for all** $\lambda \in L_{f,R}$ **do**
      **for all** $y \in Y$ **do**
        **if** $y \in Y_i$ **then**
          **replace** $r_{y(Y)}^{s_1 s_2} \in \lambda$ **with** $r_{y(Y_i)}^{s_1 s_2}$ **for all** $s_1 s_2$
        **else**
          **replace** $r_{y(Y)}^{s_1 s_2} \in \lambda$ **with** $\epsilon$ **for all** $s_1 s_2$
        **end if**
      **end for**
      $L_{f,R_i} \leftarrow L_{f,R_i} \cup \{\lambda\}$
    **end for**
  **end for**
  $L_{F_i,R_i} \leftarrow \{L_{f,R_i}\}_{f \in F_i}$
```

BFS. Both algorithms have additional work at each point in the search space explored, to compute the corresponding submodel, compute the local fault language, and check diagnosability. In an overall complexity analysis this appears as an additional factor that is not relevant in comparing the performance of BFS and the greedy algorithm.

Example 21. We apply this algorithm to the three-tank system previously described. The system is split into three diagnosis submodels, $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, where for $\mathcal{S}_1$, $F_1 = \{A_1^-, Re_1^+, Re_{12}^+\}$, $Y_1 = \{h_1\}$, for $\mathcal{S}_2$, $F_2 = \{A_2^-, Re_2^+, Re_{23}^+\}$, $Y_2 = \{h_2\}$, and for $\mathcal{S}_3$, $F_3 = \{A_3^-, Re_3^+\}$, $Y_3 = \{h_3\}$, and we use the PC-based residuals, $R_1 = \{r_{h_1(h_1)}\}$, $R_2 = \{r_{h_2(h_2)}\}$, and $R_3 = \{r_{h_3(h_3)}\}$. As a result, we have to add one residual only to the diagnosis submodels $\mathcal{S}_1$ and $\mathcal{S}_2$, and none have to be added to diagnosis submodel $\mathcal{S}_3$, because, as previously shown, $\mathcal{S}_3$ is already globally diagnosable with only $r_{h_3(h_3)}$. Using both the greedy algorithm and BFS, diagnosis submodel $\mathcal{S}_1$ gets residuals $r_{h_1(h_1,h_2)}$ and $r_{h_2(h_1,h_2)}$, and diagnosis submodel $\mathcal{S}_2$ gets residuals $r_{h_2(h_2,h_3)}$ and $r_{h_3(h_2,h_3)}$. This improves the algorithm presented in [10], because in that case, diagnosis submodel $\mathcal{S}_2$ needs three residuals, and diagnosis submodel $\mathcal{S}_3$ needs two residuals, so the size of the event-based diagnosers will be smaller.

Example 22. As another design scenario, consider now the three-tank system with $F = \{A_1^-, A_2^-, A_3^-, Re_{12}^+, Re_{23}^+\}$ and $Y = \{h_1, h_2, h_3\}$. Assume that the system is again split into three diagnosis submodels, $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, where for $\mathcal{S}_1$, $F_1 = \{A_1^-, Re_{12}^+\}$, $Y_1 = \{h_1\}$, for $\mathcal{S}_2$, $F_2 = \{A_2^-, Re_{23}^+\}$, $Y_2 = \{h_2\}$, and for $\mathcal{S}_3$, $F_3 = \{A_3^-\}$, $Y_3 = \{h_3\}$. If we use the PC-based residuals, $R_1 = \{r_{h_1(h_1)}\}$, $R_2 = \{r_{h_2(h_2)}\}$, and $R_3 = \{r_{h_3(h_3)}\}$, we see that all three diagnosis submodels, $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, are globally diagnosable. This is clear from the set of fault signatures obtained using these residuals, shown in Table 2. The PCs decouple the diagnosis submodels to the extent that only the Re_{ij}^+ faults affect multiple diagnosis submodels, and the effects they produce are unique. Hence, global diagnosability is satisfied and the diagnosis submodel design algorithm is not needed. For this design scenario, our PC-based approach also improves over the approach in [10], because in that case, diagnosis submodel $\mathcal{S}_1$ needs two residuals, and diagnosis submodel $\mathcal{S}_2$ also needs two residuals, so the size of the event-based diagnosers will be reduced and the search process will be completely avoided.

As seen in the examples, by using PCs the design approach developed in this paper improves upon the design results in [10]. Moreover, in [10], each diagnosis submodel uses the global model for residual generation. In the approach developed in this paper, however, each diagnosis submodel needs only a submodel for residual generation. So, residual generation could be implemented in a distributed way and will be more efficient.

7. Diagnoser Implementation

In this section, we describe the construction of the event-based diagnosers for fault isolation, built from the designed diagnosis submodels. An event-based diagnoser implements a diagnosis submodel, and we describe a procedure for transforming a diagnosis submodel into an event-based diagnoser. We then show that if there is a correspondence between diagnosability of a diagnosis submodel and unique isolation of faults by its diagnoser, and that if the diagnosis submodel is globally diagnosable, this unique isolation result will be globally correct.

The goal of an event-based diagnoser for a set of faults and residuals is to determine which faults are consistent with the observed sequence of residual deviation events. We define a *diagnoser* in our framework [9] as follows.

Definition 24 (Diagnoser). A *diagnoser* for a fault set F and residual set R is a tuple $\mathcal{D}_{F,R} = (S, s_0, \Sigma, \delta, A, D, G)$ where S is a set of states, $s_0 \in S$ is an initial state, Σ is a set of events, $\delta : S \times \Sigma \rightarrow S$ is a transition function, $A \subseteq S$ is a set of accepting states, $D \subseteq 2^F$ is a set of diagnosis sets, and $G : S \rightarrow D$ is a diagnosis map.

A diagnoser is a finite automaton extended by a set of diagnosis sets and a diagnosis map. As with fault models, events correspond to residual deviations. From the current state, a residual deviation event causes a transition to a new state. The diagnosis set for that new state, defined by the diagnosis map, represents the set of faults that are consistent with the sequence of events seen up to the current point in time. Under the single fault assumption, the space of possible diagnosis sets is the power set of faults, 2^F. We denote the diagnoser result (a diagnosis set) for some trace λ as $\mathcal{D}_{F,R}(\lambda)$, which is $G(s)$ for the s reached by λ. If there is no state corresponding to that trace, the diagnoser will block, and the result is $\varnothing$. Accepting states correspond to fault isolation results.

We want to construct only *correct* diagnosers, i.e., ones that faithfully capture the corresponding fault language and correctly define the diagnosis map.

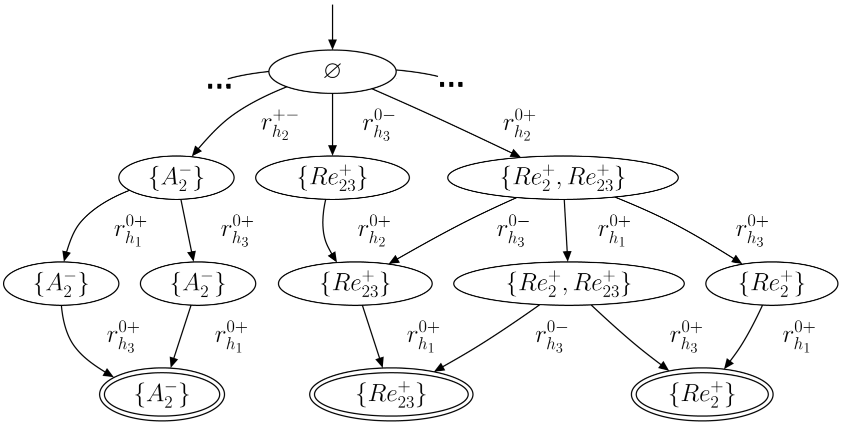


Figure 8: Fragment of the three-tank system global diagnoser for $F = \{A_2^-, Re_2^+, Re_{23}^+\}$ and $R = \{r_{h_1}, r_{h_2}, r_{h_3}\}$.

Definition 25 (Correctness)**.** A diagnoser $\mathcal{D}_{F,R}$ is *correct*, if (*i*) there is a corresponding $s \in S$ only for prefixes of traces in $L_{F,R}$, (*ii*) for all $f \in F$ and $s \in S$, $f \in G(s)$ if and only if there is a $\lambda \sqsubseteq \lambda_{f,R} \in L_{F,R}$ that reaches s, and (*iii*) for all $\lambda_{f,R} \in L_{F,R}$, the state reached by $\lambda_{f,R}$, is in A.

That is, a diagnoser is correct if it captures all valid traces, $G(s)$ contains the correct faults, and fault traces correspond to accepting states. Here, we denote a that a diagnoser is correct with the * superscript, e.g., $\mathcal{D}^*_{F,R}$. It is trivial to construct a correct diagnoser given a fault language; for a detailed procedure see [9].

Example 23. A fragment of the resulting centralized diagnoser for the three-tank system is given in Fig. 8, for $F = \{A_2^-, Re_2^+, Re_{23}^+\}$ and $R = \{r_{h_1}, r_{h_2}, r_{h_3}\}$. For example, consider the trace $r_{h_2}^{0+} r_{h_1}^{0+} r_{h_3}^{0+}$. After $r_{h_2}^{0+}$, two faults are consistent, Re_2^+ and Re_{23}^+, and diagnoser moves to a state where these two faults are contained in the diagnosis set. After $r_{h_1}^{0+}$, the diagnoser moves to a new state and the corresponding diagnosis set remains the same. After $r_{h_3}^{0+}$, now only Re_2^+ is consistent, and as this trace now corresponds to a fault trace, the state is an accepting state.

The same procedure to construct a correct centralized diagnoser applies to constructing a local diagnoser, as we simply use the diagnosis submodel with the local fault set F_i and local residual set R_i to obtain $\mathcal{D}^*_{F_i,R_i}$.

Example 24. The local diagnosers for the diagnosis submodel design example from the previous section are given in Fig. 9. Recall that diagnosis submodels $\mathcal{S}_1$ and $\mathcal{S}_2$ need to add one residual to become globally diagnosable, and diagnosis submodel $\mathcal{S}_3$ does not need any; whereas the centralized diagnoser needs all the global model residuals.

A diagnoser *isolates* a fault if it accepts all possible valid traces for the fault and the accepting states map to diagnosis sets containing the fault.

Definition 26 (Isolation)**.** A diagnoser $\mathcal{D}_{F,R}$ *isolates* fault $f \in F$ if $\mathcal{D}_{F,R}$ accepts all $\lambda_{f,R} \in L_{f,R}$ and for each $s \in A$ that accepts some $\lambda_{f,R}$, $f \in G(s)$.

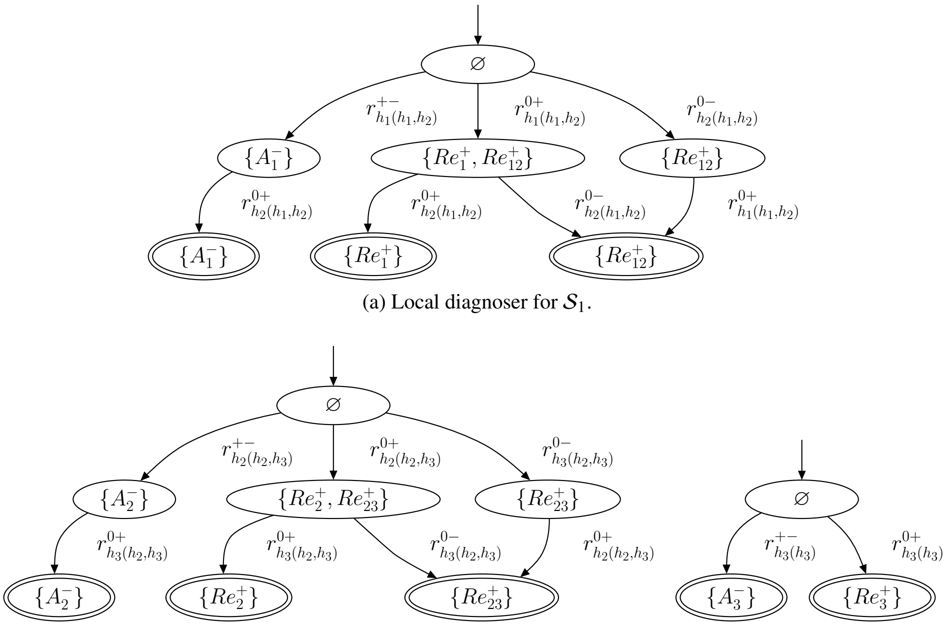
A correct diagnoser will clearly isolate all its faults; this is guaranteed by the second and third conditions of Definition 25.

Lemma 2. *If $\mathcal{D}_{F,R}$ is correct, $\mathcal{D}_{F,R}$ isolates all faults $f \in F$.*

Unique isolation corresponds to diagnosis model diagnosability. A diagnoser *uniquely isolates* a fault if each accepting state maps to the single fault.

Definition 27 (Unique Isolation)**.** A diagnoser $\mathcal{D}_{F,R}$ *uniquely isolates* fault $f \in F$ if $\mathcal{D}_{F,R}$ accepts all $\lambda_{f,R} \in L_{f,R}$ and for each $s \in A$ that accepts some $\lambda_{f,R}$, $\{f\} = G(s)$.

We can prove that a correct diagnoser $\mathcal{D}^*_{F,R}$ uniquely isolates all $f \in F$ if and only if $\mathcal{S}$ is diagnosable (adapted from [9]).



(a) Local diagnoser for $\mathcal{S}_1$.

(b) Local diagnoser for $\mathcal{S}_2$.

(c) Local diagnoser for $\mathcal{S}_3$.

Figure 9: Local diagnosers for the three-tank system for $\mathcal{S}_1$, $F_1 = \{A_1^-, Re_1^+, Re_{12}^+\}$, $R_1 = \{r_{h_1(h_1,h_2)}, r_{h_2(h_1,h_2)}\}$; $\mathcal{S}_2$, $F_2 = \{A_2^-, Re_2^+, Re_{23}^+\}$, $R_2 = \{r_{h_2(h_2,h_3)}, r_{h_3(h_2,h_3)}\}$; and $\mathcal{S}_3$, $F_3 = \{A_3^-, Re_3^+\}$, $R_3 = \{r_{h_3(h_3)}\}$.

Theorem 2. *A diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$ is diagnosable if and only if $\mathcal{D}^*_{F,R}$ uniquely isolates all $f \in F$.*

Proof. Assume some $f_i \in F$ with fault trace $\lambda_{f_i,R} \in L_{F,R}$. $\mathcal{D}^*_{F,R}$ accepts $\lambda_{f_i,R}$ and for the corresponding accepting state s^a, $f_i \in G(s^a)$ by Lemma 2 and Definition 26. Since F is diagnosable, there is no $f_j \in F$ with fault trace $\lambda_{f_j,R} \in L_{f_j,R}$ where $\lambda_{f_i,R} \sqsubseteq \lambda_{f_j,R}$. Therefore, $f_j \notin G(s^a)$. So, $G(s^a) = f_i$ and $\mathcal{D}^*_{F,R}$ uniquely isolates each $f \in F$. Assume that $\mathcal{D}^*_{F,R}$ uniquely isolates each $f \in F$. Then each possible fault trace $\lambda_{f_i,R}$ has an associated accepting state s^a, where $G(s^a) = f_i$. Thus, there cannot be some $\lambda \sqsubseteq \lambda_{f_i,R}$ for $f_i \neq f_j$ that can reach s_a, otherwise $f_j \in G(s^a)$. Therefore, $f_i \nsim f_j$, so F is diagnosable. Thus $\mathcal{S}$ is diagnosable if and only if $\mathcal{D}^*_{F,R}$ uniquely isolates each $f \in F$. □

We can apply Theorem 2 directly to a diagnosis submodel and local diagnosability, since for a diagnosis submodel, local diagnosability is equivalent to diagnosability (recall Definitions 16 and 19), to arrive at the following corollary.

Corollary 1. *A diagnosis submodel $\mathcal{S}_i$ is (locally) diagnosable if and only if $\mathcal{D}^*_{F_i,R_i}$ uniquely isolates all $f_i \in F_i$.*

This is simply a restatement of Theorem 2 as applied to a diagnosis submodel, so no proof is necessary. In Section 3, we showed that if all diagnosis submodels are globally diagnosable, then the set of diagnosis submodels is globally correct, i.e., they have the potential to generate the correct diagnosis. Given correct local diagnosers, we can show now that the correct global result will be generated.

Theorem 3. *If each diagnosis submodel in a set of n diagnosis submodels $\{\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i}) : i = 1, \ldots, n\}$, from a diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$ where $F = F_1 \cup F_2 \cup \ldots \cup F_n$, is globally diagnosable, then for all faults $f \in F$ occurring with local fault traces λ_{f,R_1}, λ_{f,R_2}, …, λ_{f,R_n}, $\bigcup_i \mathcal{D}^*_{F_i,R_i}(\lambda_{f,R_i}) = \{f\}$.*

Proof. If all $\mathcal{S}_i$ are globally diagnosable, then all $\mathcal{S}_i$ are also locally diagnosable, since $F_i \subseteq F$, by the definition of local diagnosability (Definition 19). Since they are locally diagnosable, then, by Corollary 1, $\mathcal{D}^*_{F_i,R_i}$ uniquely isolates all faults in F_i. So, if some $f \in F$ occurs, for $\mathcal{S}_i$, if $f \in F_i$, then $\mathcal{D}^*_{F_i,R_i}(\lambda_{f,R_i}) = \{f\}$ because it must uniquely isolate f. If $f \notin F_i$, then since $\mathcal{S}_i$ is globally diagnosable, f must be distinguishable from every fault in F_i using R_i, so the diagnoser will block and $\mathcal{D}^*_{F_i,R_i}(\lambda_{f,R_i}) = \varnothing$. Therefore, since f must be contained in at least one F_i, the union of all local diagnoses is always $\{f\}$. □

The local event-based diagnosers work as follows. Each local diagnoser starts in its initial state. A local residual deviation event is received by its corresponding diagnoser. If there is a matching event from the current state, a local diagnoser will follow that path to the next state, and remain active. If not, the local diagnoser will block, and its diagnosis result will be $\varnothing$. The process continues until a local diagnoser reaches an accepting state. At this point, a globally correct diagnosis is known, if this diagnosis submodel was designed to be globally diagnosable. If so, no other local diagnoser may reach an accepting state, since no two diagnosis submodels share a fault. Therefore, a globally correct diagnosis result is achieved without the use of a centralized coordinator. If the diagnosis submodels are not globally diagnosable, then two or more local diagnosers may both reach an accepting state and a coordinator would be needed to resolve the ambiguity. This holds by Theorem 3.

If a local diagnoser has not yet reached an accepting state, but has a unique diagnosis, a globally correct diagnosis result may only be declared (under the single fault assumption) if all other local diagnosers have blocked, but in this case, some kind of communication between the local diagnosers would be necessary to broadcast when they have blocked. A globally correct diagnosis result may otherwise only be declared when all residuals that are predicted to deviate for a particular fault do deviate, i.e., an accepting state is reached. These conditions correspond directly to those outlined in [8] in the absence of the event-based framework.

We can prove that the distributed approach is more scalable than the centralized approach, in terms of the diagnostic information they need to represent. We do this using the minimal representation of the diagnostic information in the form of a $|F_i||R_i|$ matrix to capture the fault signatures and $|F_i|$ matrices of size $|R|^2$ to represent the residual orderings (for each fault, residual orderings can be compactly represented in an Boolean $|R_i| \times |R_i|$ matrix where a 1 in entry (n, m) means there is an ordering $r_n \prec r_m$, and a 0 entry means there is not); here, the total minimum space required by a diagnoser for F_i and R_i is $|F_i|(|R_i| + |R_i|^2)$. A centralized diagnoser will require $|F|(|R| + |R|^2)$ space and the local diagnosers will require $\sum_i |F_i|(|R_i| + |R_i|^2)$ space.

Theorem 4. *The total information requirements of a distributed diagnoser design will grow more slowly than the information requirements of a centralized diagnoser design as the size of the system increases.*

Proof. Assume that a distributed diagnoser design is possible, i.e., that for each R_i, $|R_i| < |R|$ (recall that the design procedure guarantees that $|R_i| \leq |R|$, because $|Y_i| \leq |Y|$). In the worst case, $|R_i| = |R| - 1$. The centralized diagnoser requires $|F|(|R| + |R|^2)$ space, and the local diagnosers $\sum_i |F_i|(|R| - 1 + (|R| - 1)^2) = |F|(|R| - 1 + (|R| - 1)^2)$ space (recall that we partition the fault set, so $\sum_i |F_i| = |F|$). If the size of the fault set grows, i.e., $|F|$ becomes $|F| + 1$, the centralized approach requires $(|F|+1)(|R|+|R|^2)$ space and the distributed approach requires $(|F|+1)(|R|-1+(|R|-1)^2)$ space. The global approach grows by a larger factor. If the size of the measurement set grows, so R increases in size, i.e., $|R|$ becomes $|R| + 1$, then the centralized approach requires $(|F|)(|R| + 1 + (|R| + 1)^2)$ space and the distributed approach requires $(|F|)(|R| + |R|^2)$ space, and here again, the centralized approach grows by a larger factor. Therefore the distributed approach scales better than the centralized approach as the size of the system increases. □

8. Distributed Fault Identification

Once the fault has been isolated by the local event-based diagnoser as described in the previous section, the next step for our distributed diagnosis system is to perform distributed fault identification. In this section, we develop our proposal to include fault identification within the event-based distributed diagnosis framework. To do this, we combine our qualitative fault isolation approach with a quantitative fault identification approach that uses minimal parameter estimators similar to those presented in [15]. Minimal parameter estimators can perform fault identification in an efficient, distributed way. The basic idea underlying minimal parameter estimators is to use the PC submodels to solve a nonlinear optimization problem, for one parameter within the PC, based on its input and output values.

Example 25. Each PC contains a subset of the parameters in the system, and can be used to estimate any combination of these subsets of parameters. For example, PC $\mathcal{M}_{h_1}$ can be used to estimate the subset of parameters $\{A_1\}$, $\{Re_1\}$, $\{Re_{12}\}$, $\{A_1, Re_1\}$, $\{A_1, Re_{12}\}$, $\{Re_1, Re_{12}\}$, or $\{A_1, Re_1, Re_{12}\}$. However, since we work under the single fault assumption, we consider that each PC will only be used to estimate one parameter from the set Θ_i in $\mathcal{M}_{h_1}$.

We define a nonlinear optimization problem using the inputs U_i and the outputs Y_i of the PC to estimate the value of the parameter, $\theta_j \in \Theta_i$ related to the fault f_j. More formally, the parameter estimation problem using a PC, PC_i, computes the value of the unknown parameter, $\theta_j \in \Theta_i$, given the values U_i and Y_i, by solving the following nonlinear optimization problem:

$$\hat{\theta}_j = \underset{\theta_j}{\operatorname{argmin}} \sum_{k=1}^{t} (Y_i(k) - \hat{Y}_i(k))^2, \tag{1}$$

$$\hat{Y}_i(k) = e_{\mathcal{M}_{Y_i}}(U_i(k), \hat{\theta}_j), \tag{2}$$

where $e_{\mathcal{M}_{Y_i}}(U_i(k), \theta_j)$ is a local estimation function (that we call a local parameter estimator) obtained directly from the constraints of the PC $\mathcal{M}_{Y_i}$, that computes an estimation of the outputs of the submodel, $\hat{Y}_i$, based on its inputs and an estimated value of the parameter to identify, $\hat{\theta}_j$. Optimization is done by using the local parameter estimator and iteratively modifying the value of $\hat{\theta}_j$ to minimize the difference between the estimated outputs, $\hat{Y}_i$, and the measured outputs, Y_i (as indicated by Equation 1).

Given a diagnosis submodel $\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i})$, where the residual set is defined as $R_i = \{r_{y_1(Y_i)}, r_{y_2(Y_i)}, \ldots\}$, a local parameter estimator for a parameter θ_j, related to a fault $f_j \in F_i$, can be derived directly from the equations of the PC $\mathcal{M}_{Y_i}$, or from any of the constituent PCs of $\mathcal{M}_{Y_i}$ containing the parameter θ_j if $\mathcal{M}_{Y_i}$ is a multi-output PC.

Example 26. Consider the three-tank system. The diagnosis submodel design algorithm determined that diagnosis submodel $\mathcal{S}_1$ gets residuals $r_{h_1(h_1,h_2)}$ and $r_{h_2(h_1,h_2)}$, hence a local parameter estimator for this diagnosis submodel and parameter Re_{12} can be computed from either the multi-output PC $\mathcal{M}_{h_1,h_2}$ or any of its constituent PCs $\mathcal{M}_{h_1}$ and $\mathcal{M}_{h_2}$. However, a local parameter estimator for diagnosis submodel $\mathcal{S}_1$ and parameter Re_1 can only be computed from either the multi-output PC $\mathcal{M}_{h_1,h_2}$ or the constituent PC $\mathcal{M}_{h_1}$ (in this case $\mathcal{M}_{h_2}$ can not be used as local parameter estimator since parameter Re_1 is not included in $\mathcal{M}_{h_2}$).

As shown in the example, different local parameter estimators can be used to estimate the same parameter value. Each one of these local parameter estimators provides different features regarding accuracy and estimation time, and it is up to the user to choose which one of the local parameter estimators must be used, as described in [15]. On one hand, if the PC is chosen, the estimation process will be faster than using the multi-output PC, since it contains a smaller number of equations. On the other hand, if the multi-output PC is chosen, the estimation process will be slower than using the PCs, but, since it uses more measurements for the estimation, the result will likely be more accurate.

At run time, we use a time limit, and fault identification is triggered either when a unique diagnosis is obtained or when the time limit is reached [44]. If the local diagnoser reaches an accepting state, the local diagnoser has been able to uniquely isolate a fault, and the parameter estimation is run only for the isolated fault candidate to determine its fault magnitude. If the local diagnoser has not been able to reach an accepting state after a predefined amount of time (for example, when enough data for accurate fault identification are available), the fault identification is started for all the isolated candidates at that time.

9. Case Study

As a case study, we apply our distributed diagnosis framework to a simulation model of a rover testbed developed at NASA Ames Research Center [16, 17]. In this section, we first describe the model for the rover, then, we show several globally diagnosable diagnosis submodel design examples, and finally, we provide on-line diagnosis results in simulation for different fault situations.

9.1. Rover Modeling

The rover model (Fig. 10) assumes a symmetric rigid frame with four independently-driven wheels and four motors, one for each wheel. The rover is powered by four batteries connected in series. Four sensors, V_1^*, V_2^*, V_3^*, and V_4^*, measure the voltage in each one of the batteries, and another four sensors, ω_{FL}^*, ω_{FR}^*, ω_{BL}^*, and ω_{BR}^*, measure the wheel velocities.[17] The subscripts F, B, L, and R subscripts stand for "front", "back", "left" and "right", respectively.

[17] Since we are measuring state variables in the rover case study, we will use an asterisk ($*$) to denote the measured variables (those belonging to Y), and distinguish them from their equivalent state variables (those belonging to X).

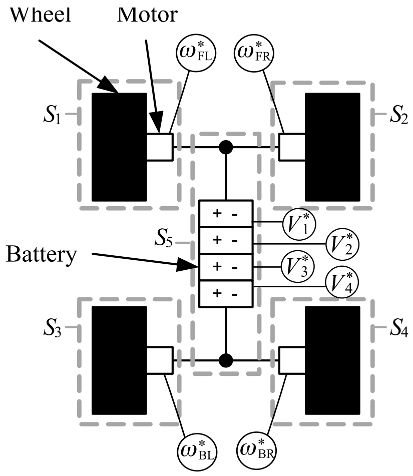


Figure 10: Rover schematic with the five diagnosis submodels.

Here, we summarize the model and refer the reader to [16, 17] for additional details. The state equations governing the dynamic behavior are given as follows. First, the wheel speeds are governed by

$$
\begin{aligned}
c_1 : \omega_{FL} &= \frac{1}{J_w}\int_{t_0}^{t}\left(k_\tau i_{FL} - \mu_{f,FL}\omega_{FL} - r_w F_{gl,FL} + r_w F_{gr}\cos\gamma\right)\cdot dt,\\
c_2 : \omega_{FR} &= \frac{1}{J_w}\int_{t_0}^{t}\left(k_\tau i_{FR} - \mu_{f,FR}\omega_{FR} - r_w F_{gl,FR} - r_w F_{gr}\cos\gamma\right)\cdot dt,\\
c_3 : \omega_{BL} &= \frac{1}{J_w}\int_{t_0}^{t}\left(k_\tau i_{BL} - \mu_{f,BL}\omega_{BL} - r_w F_{gl,BL} + r_w F_{gr}\cos\gamma\right)\cdot dt,\\
c_4 : \omega_{BR} &= \frac{1}{J_w}\int_{t_0}^{t}\left(k_\tau i_{BR} - \mu_{f,BR}\omega_{BR} - r_w F_{gl,BR} - r_w F_{gr}\cos\gamma\right)\cdot dt.
\end{aligned}
$$

The ω variables denote rotational wheel speeds, J_w denotes the wheel inertia, r_w denotes the wheel radius, and for wheel w, $\mu_{f,w}$ is a friction coefficient. The wheel forces are summarized in Fig. 11. For a wheel w, $F_{gl,w} = \mu_{gl}(v_w - v)$ is the longitudinal ground force on a wheel, where μ_{gl} is a friction coefficient, v_w is the translational wheel velocity, and v is the translational velocity of the rover frame. When there is a difference in the relative velocity between the wheel and the ground, a force is produced, due to friction, that pushes the wheel along the ground. Forces are also present resisting the rotational movement, $F_{gr,w} = \mu_r\omega_w$ for wheel w, where μ_r is a friction coefficient. Since μ_r is the same for all wheels, all the $F_{gr,w}$ forces are the same and so we drop the w subscript. For wheel w, the $k_\tau i_w$ term is for the motor torque, where i_w is the motor current and k_τ is an energy transformation gain. The angle $\gamma = \arctan\frac{l}{b}$, where l is the rover length, and b is its width.

The translational velocity v and rotational velocity ω of the rover are governed by

$$
\begin{aligned}
c_5 : v &= \frac{1}{m}\int_{t_0}^{t}\left(F_{gl,FL} + F_{gl,FR} + F_{gl,BL} + F_{gl,BR}\right)\cdot dt,\\
c_6 : \omega &= \frac{1}{J}\int_{t_0}^{t}\left(d\cos\gamma F_{gl,FR} + d\cos\gamma F_{gl,BR} - d\cos\gamma F_{gl,FL} - d\cos\gamma F_{gl,BL} - 4dF_{gr}\right)\cdot dt,
\end{aligned}
$$

where m is the rover mass, J is its rotational inertia, d is the distance from the center of the rover to each wheel.

The wheels are driven by DC motors with PID control that sets the voltages V applied to the motors. The motor

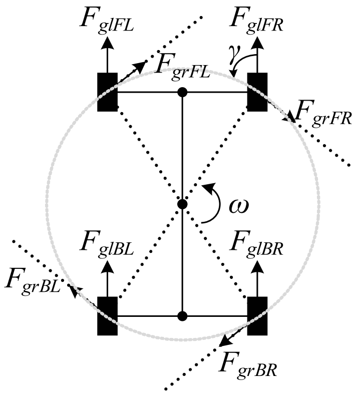


Figure 11: Rover forces.

currents i are governed by

$$
\begin{aligned}
c_7 : i_{FL} &= \frac{1}{L}\int_{t_0}^{t} (V_{FL} - i_{FL} Re_{FL} - k_\omega \omega_{FL}) \cdot dt, \\
c_8 : i_{FR} &= \frac{1}{L}\int_{t_0}^{t} (V_{FR} - i_{FR} Re_{FR} - k_\omega \omega_{FR}) \cdot dt, \\
c_9 : i_{BL} &= \frac{1}{L}\int_{t_0}^{t} (V_{BL} - i_{BL} Re_{BL} - k_\omega \omega_{BL}) \cdot dt, \\
c_{10} : i_{BR} &= \frac{1}{L}\int_{t_0}^{t} (V_{BR} - i_{BR} Re_{BR} - k_\omega \omega_{BR}) \cdot dt,
\end{aligned}
$$

where L is the motor inductance, R is the motor resistance, and k_ω is an energy transformation term.

The batteries are described by an electrical circuit equivalent model that includes a large capacitance K_b in parallel with a resistance Re_p, together in series with another resistance Re_s. The battery charge variables q_i are governed by

$$
\begin{aligned}
c_{11} : q_1 &= \int_{t_0}^{t} V_1/Re_1 - (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \cdot dt, \\
c_{12} : q_2 &= \int_{t_0}^{t} V_2/Re_2 - (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \cdot dt, \\
c_{13} : q_3 &= \int_{t_0}^{t} V_3/Re_3 - (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \cdot dt, \\
c_{14} : q_4 &= \int_{t_0}^{t} V_4/Re_4 - (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \cdot dt.
\end{aligned}
$$

As we previously mentioned, we measure the wheel speeds and the battery voltages.

$$
\begin{aligned}
c_{15} : \omega^*_{FL} &= \omega_{FL}, \\
c_{16} : \omega^*_{FR} &= \omega_{FR}, \\
c_{17} : \omega^*_{BL} &= \omega_{BL}, \\
c_{18} : \omega^*_{BR} &= \omega_{BR}, \\
c_{19} : V^*_1 &= q_1/K_{b1} - Re_1(i_{FL} + i_{FR} + i_{BR} + i_{BL}), \\
c_{20} : V^*_2 &= q_2/K_{b2} - Re_1(i_{FL} + i_{FR} + i_{BR} + i_{BL}), \\
c_{21} : V^*_3 &= q_3/K_{b3} - Re_1(i_{FL} + i_{FR} + i_{BR} + i_{BL}), \\
c_{22} : V^*_4 &= q_4/K_{b4} - Re_1(i_{FL} + i_{FR} + i_{BR} + i_{BL}),
\end{aligned}
$$

In this work we consider the following faults: increases in motor friction ($\mu^+_{f,FL}$, $\mu^+_{f,FR}$, $\mu^+_{f,BL}$, $\mu^+_{f,BR}$), capacitance loss in the batteries (K^-_{b1}, K^-_{b2}, K^-_{b3}, K^-_{b4}), and increases in electrical resistance in the motors (Re^+_{FL}, Re^+_{FR}, Re^+_{BL}, Re^+_{BR}). The fault signatures for the set of faults and measurements for the rover are listed in Table 4.[18] To improve readability, only a subset of the residual orderings is shown.

Table 4: Fault signatures for the global model of the rover.

Fault	r_{V_1}	r_{V_2}	r_{V_3}	r_{V_4}	$r_{\omega_{FL}}$	$r_{\omega_{FR}}$	$r_{\omega_{BL}}$	$r_{\omega_{BR}}$	Residual Orderings
$\mu^+_{f,FL}$	0−	0−	0−	0−	0−	0−	0⋆	0−	$r_{\omega_{FL}} \prec r_{V_1}, r_{\omega_{FL}} \prec r_{V_4}$
$\mu^+_{f,FR}$	0−	0−	0−	0−	0−	0−	0−	0⋆	$r_{\omega_{FR}} \prec r_{V_1}, r_{\omega_{FR}} \prec r_{V_4}$
$\mu^+_{f,BL}$	0−	0−	0−	0−	0⋆	0−	0−	0−	$r_{\omega_{BL}} \prec r_{V_1}, r_{\omega_{BL}} \prec r_{V_4}$
$\mu^+_{f,BR}$	0−	0−	0−	0−	0−	0⋆	0−	0−	$r_{\omega_{BR}} \prec r_{V_1}, r_{\omega_{BR}} \prec r_{V_4}$
K^-_{b1}	+0	00	00	00	00	00	00	00	$r_{V_1} \prec r_{V_2}, r_{V_1} \prec r_{V_3}$
K^-_{b2}	00	+0	00	00	00	00	00	00	$r_{V_2} \prec r_{V_1}, r_{V_2} \prec r_{V_3}$
K^-_{b3}	00	00	+0	00	00	00	00	00	$r_{V_3} \prec r_{V_1}, r_{V_3} \prec r_{V_2}$
K^-_{b4}	00	00	00	+0	00	00	00	00	$r_{V_4} \prec r_{V_1}, r_{V_4} \prec r_{V_2}$
Re^+_{FL}	0+	0+	0+	0+	0−	0−	0⋆	0−	$r_{\omega_{FL}} \prec r_{\omega_{FR}}, r_{\omega_{FL}} \prec r_{\omega_{BL}}$
Re^+_{FR}	0+	0+	0+	0+	0−	0−	0−	0⋆	$r_{\omega_{FR}} \prec r_{\omega_{FL}}, r_{\omega_{FR}} \prec r_{\omega_{BL}}$
Re^+_{BL}	0+	0+	0+	0+	0⋆	0−	0−	0−	$r_{\omega_{BL}} \prec r_{\omega_{FL}}, r_{\omega_{BL}} \prec r_{\omega_{FR}}$
Re^+_{BR}	0+	0+	0+	0+	0−	0⋆	0−	0−	$r_{\omega_{BR}} \prec r_{\omega_{FL}}, r_{\omega_{BR}} \prec r_{\omega_{FR}}$

We have a set of eight PCs in the rover, one for each measurement, as generated by Algorithm 1. The PCs are defined as follows:

$$\begin{aligned}
\mathcal{M}_{V_1} &= (\{\{q_1, i_{FL}, i_{FR}, i_{BL}, i_{BR}\} \cup \{K_{b1}, Re_{FL}, Re_{FR}, Re_{BL}, Re_{BR}\} \cup \{v_{FL}, v_{FR}, v_{BL}, v_{BR}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}\} \cup \{V^*_1\}\},\\
&\quad \{c_7, c_8, c_9, c_{10}, c_{11}, c_{15}, c_{16}, c_{17}, c_{18}, c_{19}\}),\\
\mathcal{M}_{V_2} &= (\{\{q_2, i_{FL}, i_{FR}, i_{BL}, i_{BR}\} \cup \{K_{b2}, Re_{FL}, Re_{FR}, Re_{BL}, Re_{BR}\} \cup \{v_{FL}, v_{FR}, v_{BL}, v_{BR}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}\} \cup \{V^*_2\}\},\\
&\quad \{c_7, c_8, c_9, c_{10}, c_{12}, c_{15}, c_{16}, c_{17}, c_{18}, c_{20}\}),\\
\mathcal{M}_{V_3} &= (\{\{q_3, i_{FL}, i_{FR}, i_{BL}, i_{BR}\} \cup \{K_{b3}, Re_{FL}, Re_{FR}, Re_{BL}, Re_{BR}\} \cup \{v_{FL}, v_{FR}, v_{BL}, v_{BR}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}\} \cup \{V^*_3\}\},\\
&\quad \{c_7, c_8, c_9, c_{10}, c_{13}, c_{15}, c_{16}, c_{17}, c_{18}, c_{21}\}),\\
\mathcal{M}_{V_4} &= (\{\{q_4, i_{FL}, i_{FR}, i_{BL}, i_{BR}\} \cup \{K_{b4}, Re_{FL}, Re_{FR}, Re_{BL}, Re_{BR}\} \cup \{v_{FL}, v_{FR}, v_{BL}, v_{BR}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}\} \cup \{V^*_4\}\},\\
&\quad \{c_7, c_8, c_9, c_{10}, c_{14}, c_{15}, c_{16}, c_{17}, c_{18}, c_{22}\}),\\
\mathcal{M}_{\omega_{FL}} &= (\{\{\omega_{FL}, i_{FL}, w, v\} \cup \{\mu_{f,FL}, Re_{FL}\} \cup \{v_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}\} \cup \{\omega^*_{FL}\}\}, \{c_1, c_5, c_6, c_7, c_{15}, c_{16}, c_{17}, c_{18}\}),\\
\mathcal{M}_{\omega_{FR}} &= (\{\{\omega_{FR}, i_{FR}, w, v\} \cup \{\mu_{f,FR}, Re_{FR}\} \cup \{v_{FR}, \omega^*_{FL}, \omega^*_{BL}, \omega^*_{BR}\} \cup \{\omega^*_{FR}\}\}, \{c_2, c_5, c_6, c_8, c_{15}, c_{16}, c_{17}, c_{18}\}),\\
\mathcal{M}_{\omega_{BL}} &= (\{\{\omega_{BL}, i_{BL}, w, v\} \cup \{\mu_{f,BL}, Re_{BL}\} \cup \{v_{BL}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BR}\} \cup \{\omega^*_{BL}\}\}, \{c_3, c_5, c_6, c_9, c_{15}, c_{16}, c_{17}, c_{18}\}),\\
\mathcal{M}_{\omega_{BR}} &= (\{\{\omega_{BR}, i_{BR}, w, v\} \cup \{\mu_{f,BR}, Re_{BR}\} \cup \{v_{BR}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}\} \cup \{\omega^*_{BR}\}\}, \{c_4, c_5, c_6, c_{10}, c_{15}, c_{16}, c_{17}, c_{18}\}).
\end{aligned}$$

Fault signatures for the minimal submodels are shown in Table 5. As it happens with the three-tank system, the PCs are able to decouple the faults from the measurements, so that each residual is only affected by a subset of the faults.

Next, we show the application of our distributed diagnosis framework to the rover. First, we show different design scenarios and compare the design obtained with the proposed approach (using PCs) against the design obtained using the approach in [10] (using the global system model). Then, we apply our unified distributed diagnosis framework to several fault scenarios to illustrate on-line fault diagnosis capabilities of the approach.

9.2. Globally Diagnosable Diagnosis Submodel Design Experiments

Let us consider the initial rover configuration with $F = \{\mu^+_{f,FL}, \mu^+_{f,FR}, \mu^+_{f,BL}, \mu^+_{f,BR}, K^-_{b1}, K^-_{b2}, K^-_{b3}, K^-_{b4}, Re^+_{FL}, Re^+_{FR}, Re^+_{BL}, Re^+_{BR}\}$ and $Y = \{V^*_1, V^*_2, V^*_3, V^*_4, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}\}$. Now, assume that the system is split into five diagnosis

[18] We use an ($\star$) in Table 4 to represent a qualitative ambiguity in the result of our fault signature generation algorithm. The $\star$ can manifest either as a + or a − on the real system, e.g., the 0⋆ in the table is a shorthand that represents two distinct signatures, 0+ and 0−.

Table 5: Fault signatures for the set of minimal submodels of the rover.

Fault	$r_{V_1(V_1)}$	$r_{V_2(V_2)}$	$r_{V_3(V_3)}$	$r_{V_4(V_4)}$	$r_{\omega_{FL}(\omega_{FL})}$	$r_{\omega_{FR}(\omega_{FR})}$	$r_{\omega_{BL}(\omega_{BL})}$	$r_{\omega_{BR}(\omega_{BR})}$	Residual Orderings
$\mu^+_{f,FL}$	00	00	00	00	0−	00	00	00	∅
$\mu^+_{f,FR}$	00	00	00	00	00	0−	00	00	∅
$\mu^+_{f,BL}$	00	00	00	00	00	00	0−	00	∅
$\mu^+_{f,BR}$	00	00	00	00	00	00	00	0−	∅
K^-_{b1}	+0	00	00	00	00	00	00	00	∅
K^-_{b2}	00	+0	00	00	00	00	00	00	∅
K^-_{b3}	00	00	+0	00	00	00	00	00	∅
K^-_{b4}	00	00	00	+0	00	00	00	00	∅
Re^+_{FL}	0+	0+	0+	0+	0−	00	00	00	∅
Re^+_{FR}	0+	0+	0+	0+	00	0−	00	00	∅
Re^+_{BL}	0+	0+	0+	0+	00	00	0−	00	∅
Re^+_{BR}	0+	0+	0+	0+	00	00	00	0−	∅

submodels (see Fig. 10): $\mathcal{S}_1$ related to the front-left wheel components, $\mathcal{S}_2$ for the front-right wheel components, $\mathcal{S}_3$ for the back-left wheel components, $\mathcal{S}_4$ for the back-right wheel components, and $\mathcal{S}_5$ for the components of the battery. Y_i and F_i subsets for each diagnosis submodel are described as follows:

$$\begin{aligned}
\mathcal{S}_1 &\Rightarrow F_1 = \{\mu^+_{f,FL}, Re^+_{FL}\}; Y_1 = \{\omega^*_{FL}\},\\
\mathcal{S}_2 &\Rightarrow F_2 = \{\mu^+_{f,FR}, Re^+_{FR}\}; Y_2 = \{\omega^*_{FR}\},\\
\mathcal{S}_3 &\Rightarrow F_3 = \{\mu^+_{f,BL}, Re^+_{BL}\}; Y_3 = \{\omega^*_{BL}\},\\
\mathcal{S}_4 &\Rightarrow F_4 = \{\mu^+_{f,BR}, Re^+_{BR}\}; Y_4 = \{\omega^*_{BR}\},\\
\mathcal{S}_5 &\Rightarrow F_5 = \{K^-_{b1}, K^-_{b2}, K^-_{b3}, K^-_{b4}\}; Y_5 = \{V^*_1, V^*_2, V^*_3, V^*_4\}.
\end{aligned}$$

First, let us assume we use the $\mathcal{M}$-based residuals, hence $R_1 = \{r_{\omega_{FL}}\}$, $R_2 = \{r_{\omega_{FR}}\}$, $R_3 = \{r_{\omega_{BL}}\}$, $R_4 = \{r_{\omega_{BR}}\}$, and $R_5 = \{r_{V_1}, r_{V_2}, r_{V_3}, r_{V_4}\}$. Analyzing global diagnosability from Table 4, we see that only diagnosis submodel $\mathcal{S}_5$ is globally diagnosable, since faults K^-_{b1} to K^-_{b4} can be distinguished from the rest of the faults in the system, since the effects produced by the faults (+0 for all of them) are different from the rest of effects produced in sensors V_1 to V_4 for the rest of the faults considered. None of the remaining diagnosis submodels is globally diagnosable, hence we have to add new residuals to each of those diagnosis submodels in order to satisfy the global diagnosability design constraints. In particular, diagnosis submodels $\mathcal{S}_1$ to $\mathcal{S}_4$ need to merge three residuals each. Table 6 shows the design results using $\mathcal{M}$-based residuals as in [10]. The table indicates the initial residual sets for each diagnosis submodel and the additional residuals needed to make the diagnosis submodels globally diagnosable.

Table 6: Design results using $\mathcal{M}$-based residuals for the first design scenario.

Diagnosis submodel	Diagnosable?	Initial R_i set	Additional residuals
$\mathcal{S}_1$	No	$r_{\omega_{FL}}$	$r_{\omega_{BL}}, r_{V_1}$
$\mathcal{S}_2$	No	$r_{\omega_{FR}}$	$r_{\omega_{BL}}, r_{V_1}$
$\mathcal{S}_3$	No	$r_{\omega_{BL}}$	$r_{\omega_{BR}}, r_{V_1}$
$\mathcal{S}_4$	No	$r_{\omega_{BR}}$	$r_{\omega_{BL}}, r_{V_1}$
$\mathcal{S}_5$	Yes	$r_{V_1}, r_{V_2}, r_{V_3}, r_{V_4}$	∅

Now assume that we use the PC-based residuals, i.e., $R_1 = \{r_{\omega_{FL}(\omega_{FL})}\}$, $R_2 = \{r_{\omega_{FR}(\omega_{FR})}\}$, $R_3 = \{r_{\omega_{BL}(\omega_{BL})}\}$, $R_4 = \{r_{\omega_{BR}(\omega_{BR})}\}$, and $R_5 = \{r_{V_1(V_1)}, r_{V_2(V_2)}, r_{V_3(V_3)}, r_{V_4(V_4)}\}$. If we look at Table 5, we see that diagnosis submodel $\mathcal{S}_5$ is globally diagnosable, and the remaining diagnosis submodels are not globally diagnosable, hence they all need to run the design algorithm to find the minimal submodels that need to be merged to get all diagnosis submodels globally diagnosable. Using both the greedy algorithm and BFS, we obtained that only two PCs need to be merged with the to make diagnosis submodels $\mathcal{S}_1$ to $\mathcal{S}_4$ globally diagnosable. Table 7 shows the design results using PC-based residuals.

Table 7: Design results using PC-based residuals for the first design scenario.

Diagnosis submodel	Diagnosable?	Initial R_i set	Additional residuals
$\mathcal{S}_1$	No	$r_{\omega_{FL}(\omega_{FL})}$	$r_{V_4(V_4)}$
$\mathcal{S}_2$	No	$r_{\omega_{FR}(\omega_{FR})}$	$r_{V_4(V_4)}$
$\mathcal{S}_3$	No	$r_{\omega_{BL}(\omega_{BL})}$	$r_{V_4(V_4)}$
$\mathcal{S}_4$	No	$r_{\omega_{BR}(\omega_{BR})}$	$r_{V_4(V_4)}$
$\mathcal{S}_5$	Yes	$r_{V_1(V_1)}, r_{V_2(V_2)}, r_{V_3(V_3)}, r_{V_4(V_4)}$	$\varnothing$

As shown in Tables 6 and 7, using PC-based residuals the design results are improved in two different ways: (1) the size of the local diagnosers for diagnosis submodels $\mathcal{S}_1$ to $\mathcal{S}_4$ after the design process is smaller than the size of the diagnosers using $\mathcal{M}$-based residuals (using PC-based residuals, only one additional residual, instead of two, is needed); and (2), PC-based residuals allow to have a fully distributed approach, not only regarding local diagnosers, but also regarding the residual generators (with $\mathcal{M}$-based residuals the diagnosis process will be distributed, but the residual computation will still be centralized).

Another additional advantage of our approach is that, it would be able to provide globally diagnosable diagnosis submodel with almost no design effort. As an example of this situation, consider now the rover with the same set of measurements Y, but a reduced set of fault candidates: $F = \{\mu^+_{f,FL}, \mu^+_{f,FR}, \mu^+_{f,BL}, \mu^+_{f,BR}, K^-_{b1}, K^-_{b2}, K^-_{b3}, K^-_{b4}\}$. Diagnosis submodels are similar to the first scenario:

$$\begin{aligned}
&\mathcal{S}_1 \Rightarrow F_1 = \{\mu^+_{f,FL}\}; Y_1 = \{\omega^*_{FL}\},\\
&\mathcal{S}_2 \Rightarrow F_2 = \{\mu^+_{f,FR}\}; Y_2 = \{\omega^*_{FR}\},\\
&\mathcal{S}_3 \Rightarrow F_3 = \{\mu^+_{f,BL}\}; Y_3 = \{\omega^*_{BL}\},\\
&\mathcal{S}_4 \Rightarrow F_4 = \{\mu^+_{f,BR}\}; Y_4 = \{\omega^*_{BR}\},\\
&\mathcal{S}_5 \Rightarrow F_5 = \{K^-_{b1}, K^-_{b2}, K^-_{b3}, K^-_{b4}\}; Y_5 = \{V^*_1, V^*_2, V^*_3, V^*_4\}.
\end{aligned}$$

After analyzing global diagnosability for this example, we find that when $\mathcal{M}$-based residuals are used, design for diagnosis submodels $\mathcal{S}_1$ to $\mathcal{S}_4$ is still needed, but if PC-based residuals are used, all diagnosis submodels are globally diagnosable, and no further design is needed. Tables 8 and 9 show these results.

Table 8: Design results using $\mathcal{M}$-based residuals for the second design scenario.

Diagnosis submodel	Diagnosable?	Initial R_i set	Additional residuals
$\mathcal{S}_1$	No	$r_{\omega_{FL}}$	$r_{\omega_{BL}}$
$\mathcal{S}_2$	No	$r_{\omega_{FR}}$	$r_{\omega_{BL}}$
$\mathcal{S}_3$	No	$r_{\omega_{BL}}$	$r_{\omega_{BR}}$
$\mathcal{S}_4$	No	$r_{\omega_{BR}}$	$r_{\omega_{BL}}$
$\mathcal{S}_5$	Yes	$r_{V_1}, r_{V_2}, r_{V_3}, r_{V_4}$	$\varnothing$

Table 9: Design results using PC-based residuals for the second design scenario.

Diagnosis submodel	Diagnosable?	Initial R_i set	Additional residuals
$\mathcal{S}_1$	Yes	$r_{\omega_{FL}(\omega_{FL})}$	$\varnothing$
$\mathcal{S}_2$	Yes	$r_{\omega_{FR}(\omega_{FR})}$	$\varnothing$
$\mathcal{S}_3$	Yes	$r_{\omega_{BL}(\omega_{BL})}$	$\varnothing$
$\mathcal{S}_4$	Yes	$r_{\omega_{BR}(\omega_{BR})}$	$\varnothing$
$\mathcal{S}_5$	Yes	$r_{V_1(V_1)}, r_{V_2(V_2)}, r_{V_3(V_3)}, r_{V_4(V_4)}$	$\varnothing$

Having shown the design properties of our distributed framework, the next section shows the performance of the

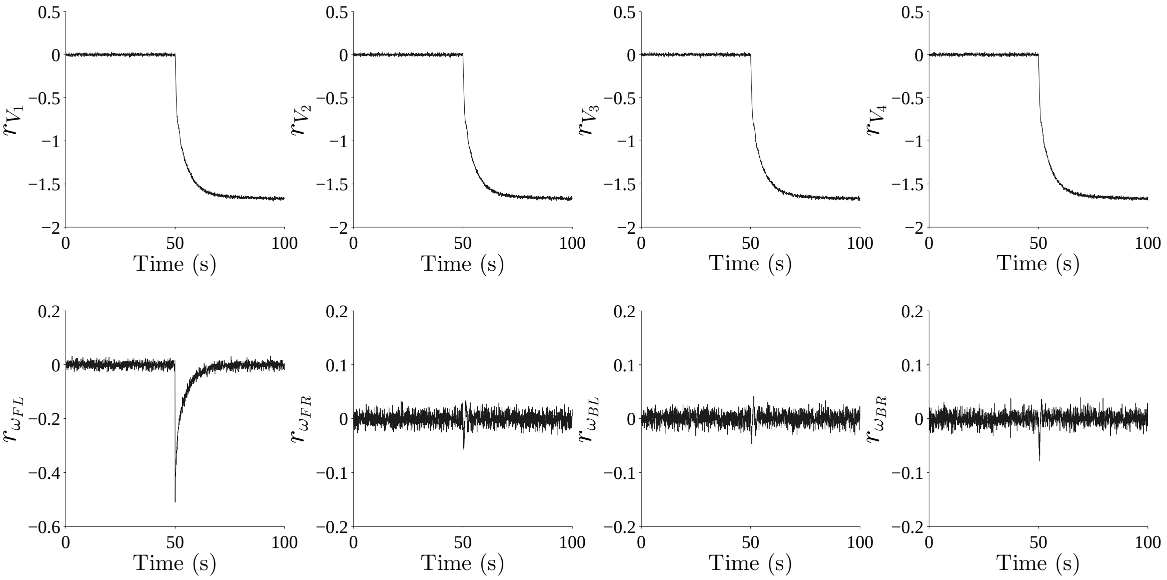


Figure 12: Residual signals for the centralized approach when a 20% friction increase in the front-left wheel, $\mu^+_{f,FL}$, occurs at time 50 s.

distributed diagnosis approach using simulated data for several fault diagnosis experiments and compares the results obtained against a centralized approach.

9.3. On-line Fault Diagnosis

As a particular example to demonstrate on-line diagnosis in this framework, consider a 20% friction increase in the front-left wheel of the rover, $\mu^+_{f,FL}$, occurring at time 50 s. Fig. 12 shows the plots of the residuals for this fault when the centralized approach is used. At time 50.05 s, a decrease in residual $r_{\omega_{FL}}$ is detected by the centralized diagnoser, and the diagnoser moves to a state where faults $\mu^+_{f,FL}$ and Re^+_{FL} are generated as the initial set of candidates. Later, at 50.15 s, decreases in residuals r_{V_1}, r_{V_2}, r_{V_3}, and r_{V_4} are detected, and the diagnoser moves to a state with diagnosis $\{\mu^+_{f,FL}\}$. Hence, the hypothesized path to the state with $\{Re^+_{FL}\}$ is eliminated and the diagnosis is confirmed as $\{\mu^+_{f,FL}\}$. At this point, the fault identification for $\mu^+_{f,FL}$ is triggered, computing the fault magnitude with an accuracy of 97.22%.

Now, let us consider the distributed diagnosis approach for the same fault and the first diagnosis submodel configuration (Table 7). Fig. 13 shows the plots of the residuals of the local diagnoser for diagnosis submodel $\mathcal{S}_1$ ($r_{\omega_{FL}(\omega_{FL},V_4)}$ and $r_{V_4(\omega_{FL},V_4)}$). At time 50.05 s, a decrease in residual $r_{\omega_{FL}(\omega_{FL},V_4)}$ is detected in the local diagnoser for $\mathcal{S}_1$, which moves to a state with diagnosis $\{\mu^+_{f,FL}, Re^+_{FL}\}$ as the initial set of candidates. No deviations are detected in the residuals for the rest of the local diagnosers. At 50.20 s, a decrease in $r_{V_4(\omega_{FL},V_4)}$ is detected and the diagnoser moves to a state with diagnosis $\{\mu^+_{f,FL}\}$. Since the diagnoser has reached to an accepting state, a global diagnosis has been achieved. Similarly to the centralized approach, the fault identification is triggered, determining the fault magnitude with an accuracy of 99.01%.

To make a more detailed study regarding the quality of our approach, we ran a set of several diagnosis experiments for each one of the faults in the system and considered different fault magnitudes (here we only show results for 5%, 10%, and 20% faults). Each run covered 100 s, and the faults were introduced at time t = 50 s. Gaussian measurement noise with a power level set to 5% of the average signal power was added to each measurement. For each experiment we randomly varied the noise signal.

Table 10 shows the results we obtained, comparing the performance of the centralized diagnosis approach against our distributed diagnosis approach for the first design scenario (Table 7). The metrics in Table 10 consist of the mean time in seconds to detect faults M_{fd}, the mean time in seconds to isolate faults M_{fi}, and the mean of the accuracy and standard deviation in the fault identification M_{da} and M_{std}, respectively.

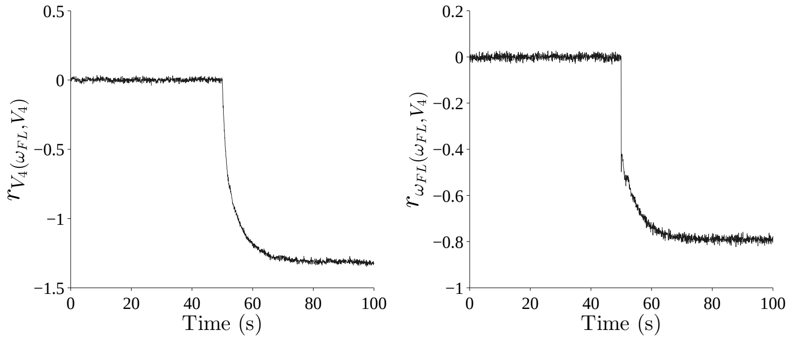


Figure 13: Residual signals for the local diagnoser of diagnosis submodel $\mathcal{S}_1$ when a 20% friction increase in the front-left wheel, $\mu^+_{f,FL}$, occurs at time 50 s.

For fault detection, a threshold based on the Z-test is computed, and to account for modeling error, an additional error term E is added to the threshold [43]. When the absolute value of the mean residual value over a small window is over this combined threshold, a fault is detected. From Table 10, it is clear that both approaches perform in a very similar way. In 66.6% of the experiments, the average detection time was the same for both approaches, and for the rest 33.3% of the experiments, the centralized approach behaved a little bit better than the distributed. This is due to the PCs decomposition, which partition the system model by using sensors as inputs, which are noisy, and this corrupts the predictions with noise. Since the detectors were tuned to avoid false alarms, the detectors for the distributed approach could not be tuned to be as sensitive as for the centralized approach. However, the differences were not significant.

Fault isolation works, as explained in Section 7, by capturing the qualitative values of residual deviations [7]. Results in Table 10 show very similar isolation times for both approaches, except for the capacitance in the batteries (K^-_b) fault experiments. In these cases, the distributed approach was able to isolate faults in only 0.05 s, while the centralized approach took up to 0.5 s. This is explained by the enhanced diagnosability obtained by the decoupling of PCs. However, as it happens with fault detection times, the differences are not significant in general, and we can consider that both approaches obtain equivalent fault isolation results.

Fault identification works as detailed in Section 8. In our particular implementation, we used non-linear least squares (LS) as the optimization algorithm and 0.5 s for the time limit. The fault candidate with the smallest LS error will be selected as the correct fault candidate. Experimental results show that, for most of the experiments, the performance is better for the distributed approach. The smaller size of the local parameter estimators, together with the reduced number of measurements used by each one, makes the distributed identification approach to converge faster than the centralized approach. In particular, the mean value of accuracy in the fault identification for the centralized approach for motor friction (μ^+_f) and resistance in the motors (R^+) faults, was 97.5% and 94.7%, respectively. Meanwhile, the mean value of accuracy in the fault identification for the same faults but for the distributed approach was 99.2% and 96.8%, respectively. For capacitance in the batteries (K^-_b) faults, both approaches achieved practically the same accuracy, 99.8% for the centralized approach against 99.6% for the distributed approach. Regarding standard deviation, in both cases the results are very accurate, although the results for the centralized approach are slightly better in most of the experiments than the results for the distributed approach. Similarly to the fault detection results, this was caused by the noisy sensors used as input by the distributed approach.

10. Related Work

Researchers have developed several decentralized and distributed diagnosis schemes in the past, mostly in the discrete-event framework [6, 36, 37, 45]. Distributed schemes, e.g., [46], unlike decentralized schemes, such as [37], do not make use of the global system model; instead, they use subsystem models for diagnosis, and the local diagnosers for each diagnosis submodel communicate their diagnosis results to each other to obtain the global solution. Decentralized diagnosis approaches, e.g., [37], typically start with a global system model to generate the local diagnosers

Table 10: Centralized and Distributed Diagnosis results for 5%, 10%, and 20% faults for the rover. M_{td} is the mean time to detect faults, M_{ti} is the mean time to isolate faults, and M_{da} and M_{std} are the mean of the accuracy and standard deviation in the fault identification.

		Centralized				**Distributed**			
Fault	Magnitude	$M_{td}(s)$	$M_{ti}(s)$	$M_{da}(\%)$	M_{std}	$M_{td}(s)$	$M_{ti}(s)$	$M_{da}(\%)$	M_{std}
$\mu^+_{f,FL}$	5%	0.15	0.22	98.04	0.06	0.15	0.32	98.91	0.13
	10%	0.10	0.15	97.50	0.09	0.10	0.25	99.65	0.18
	20%	0.05	0.12	96.82	0.14	0.05	0.20	99.36	0.27
$\mu^+_{f,FR}$	5%	0.15	0.21	98.15	0.06	0.15	0.31	98.92	0.13
	10%	0.10	0.17	97.51	0.09	0.10	0.24	99.24	0.19
	20%	0.05	0.13	97.00	0.14	0.05	0.20	99.23	0.27
$\mu^+_{f,BL}$	5%	0.15	0.20	97.96	0.06	0.15	0.31	98.94	0.12
	10%	0.10	0.16	97.41	0.09	0.10	0.25	99.43	0.18
	20%	0.05	0.15	96.95	0.14	0.05	0.20	98.86	0.28
$\mu^+_{f,BR}$	5%	0.15	0.20	97.97	0.06	0.15	0.29	98.82	0.13
	10%	0.10	0.15	97.49	0.09	0.10	0.24	99.25	0.18
	20%	0.05	0.15	96.96	0.14	0.05	0.20	99.45	0.29
K^-_{b1}	5%	0.05	0.50	99.55	0.00043	0.05	0.05	99.24	0.00053
	10%	0.05	0.50	99.82	0.00042	0.05	0.05	99.71	0.00047
	20%	0.05	0.50	99.97	0.00031	0.05	0.05	99.89	0.00035
K^-_{b2}	5%	0.05	0.50	99.58	0.00043	0.05	0.05	99.20	0.00055
	10%	0.05	0.50	99.90	0.00039	0.05	0.05	99.70	0.00039
	20%	0.05	0.50	99.93	0.00031	0.05	0.05	99.84	0.00043
K^-_{b3}	5%	0.05	0.50	99.63	0.00044	0.05	0.05	99.06	0.00050
	10%	0.05	0.50	99.77	0.00040	0.05	0.05	99.57	0.00044
	20%	0.05	0.50	99.94	0.00031	0.05	0.05	99.89	0.00032
K^-_{b4}	5%	0.05	0.50	99.70	0.00048	0.05	0.05	99.15	0.00051
	10%	0.05	0.50	99.82	0.00039	0.05	0.05	99.54	0.00047
	20%	0.05	0.50	99.97	0.00033	0.05	0.05	99.81	0.00033
Re^+_{FL}	5%	0.19	0.67	92.80	0.29	0.36	0.66	98.62	0.28
	10%	0.14	0.51	94.54	0.49	0.23	0.55	96.70	0.54
	20%	0.10	0.45	92.82	0.99	0.16	0.46	95.92	1.30
Re^+_{FR}	5%	0.17	0.67	94.88	0.28	0.28	0.59	95.38	0.27
	10%	0.14	0.53	95.96	0.50	0.22	0.53	97.04	0.54
	20%	0.08	0.45	94.66	0.94	0.16	0.47	96.01	1.32
Re^+_{BL}	5%	0.19	0.67	94.06	0.29	0.29	0.59	96.41	0.27
	10%	0.14	0.54	94.92	0.49	0.23	0.53	97.56	0.52
	20%	0.10	0.45	95.63	0.96	0.18	0.51	97.07	1.39
Re^+_{BR}	5%	0.19	0.65	91.16	0.28	0.33	0.63	97.24	0.27
	10%	0.14	0.54	95.33	0.49	0.22	0.53	96.12	0.58
	20%	0.09	0.45	95.36	1.00	0.18	0.49	98.09	1.32

among which the diagnosis computations get distributed. Each local decentralized diagnoser makes their diagnosis decision based on only a subset of observable events, and they communicate these decisions to other diagnosers, or to a centralized coordinator, which uses the global model to generate globally consistent diagnosis solutions. The level of coordination required between the local diagnosers depends on how each local diagnoser is designed. In [37], three coordinated decentralized protocols for diagnosis are presented. Coordination is necessary in the first and second protocols to generate the correct diagnosis result, but the third protocol generates correct results without a coordinator. Our diagnosis approach is similar to this third protocol, since we do not need a central coordinator, and a failure in the local diagnoser does not affect the diagnosis capability of the other diagnosers. We attribute this property of our diagnosis algorithm to the careful design of the local diagnosers based on global diagnosability properties.

Other researchers avoid coordination between local diagnosers by representing the system as a network of communicating finite state machines, such as in [40, 47]. In these approaches, first, the observable events for each subsystem are used to generate the individual subsystem diagnoses. Then the subsystem diagnoses are merged to generate the global diagnosis result. The approach presented in [47] assumes all observable events are received in the same order that they were transmitted in. The on-line approach described in [40] does not assume the ordering of observations is preserved. Our approach avoids these problems since it works only with local observations, and does not communicate observations or results to other local diagnosers.

In [48], the authors describe an approach where each local diagnoser generates a set of local diagnoses, and then communicates with its neighbors to reduce the number of hypotheses. The graph of constraints between the fault hypotheses and the observations is partitioned to minimize communication between local diagnosers. A similar approach is presented in [46], where the partitioning is based on physical connections.

In [49], the authors also use structural model decomposition (Analytical Redundancy Relations (ARRs) [12] in their case) to propose a decentralized but coordinated architecture for distributed fault diagnosis of continuous systems. The system is decomposed into functional subsystems which are provided by the user as input. The architecture is hierarchical and composed of local diagnosers for subsystems which work with local models. The local diagnosers are made up of a set of residual generators built upon the ARR concept. Each local diagnoser has a different residual generator. The hierarchical approach means that the scheme can be replicated at different levels, and work at different levels in the hierarchy. Unlike our approach, local results need to be communicated to a central supervisor which is responsible for the fault isolation task.

One of the main differences of our framework regarding other continuous distributed diagnosis approaches is that we use minimal submodels to design local diagnosers off-line, which yields smaller size diagnosers, and allows a fully distributed approach. Then, at runtime, all the local diagnosers operate independently to generate local diagnosis results that are globally correct. Our approach does not require a coordinator, and there is no exchange of diagnosis information among the diagnosers, only the exchange of measurements.

Our event-based framework is similar to traditional discrete event systems (DES) approaches [36, 45], except that the event-based models are derived from the continuous model and capture only fault-specific dynamics in a qualitative abstraction. One difference with pure DES approaches for diagnosis is that we use the local event-based diagnosis to perform fault isolation, after a robust fault detection stage, thus limiting the event analysis. Other continuous systems diagnosis approaches also use temporal information in similar ways, e.g., [50–55]. An advantage in our approach is that temporal information can be automatically generated given the equations and the structural model of the system.

11. Conclusions

In this work, we have developed a novel comprehensive framework for distributed qualitative fault diagnosis of continuous systems using structural model decomposition, where PCs are used to decouple the faults from the measurements and compute minimal submodels for distributed diagnosis. Then, the basic PC framework is extended to allow merging of PCs to design globally diagnosable diagnosis submodels. We proposed an algorithm that merges PCs (when necessary) to design local distributed diagnosis submodels based on global diagnosability. These local distributed diagnosis submodels are then used to construct local event-based distributed diagnosers. Finally, we use the model decomposition capabilities of the PCs not only to improve the design, but also to integrate within our framework a distributed fault identification approach by developing independent local parameter estimators. Our main conclusion is that using a structural model decomposition method, such as PCs, together with the existing consistency-based diagnosis approach using qualitative information, results in an event-based distributed diagnosis framework covering fault detection, isolation, and identification that scales well, and where the local diagnosers are independent at every level (no communication between them at any stage is required), do not need a central coordinator, and have no single point of failure.

Experimental results on a rover showed the improvement of the design using submodels against the approach that uses the global model of the system [10]. The decoupling obtained by the PCs improved the design process and the size of the local diagnosers, thus providing a scalable solution. Using the rover, we also ran several experiments to compare the performance of a centralized approach against our distributed approach regarding fault detection, isolation and identification. Results for different kinds of faults with several fault magnitudes have shown that the distributed approach is able to obtain similar detection and isolation results to the centralized approach, while avoiding the problems related to the centralized scheme. Regarding fault identification, results obtained using the distributed approach with the local parameter estimators showed an improvement in the accuracy of the estimation with just a small loss in precision.

Some limitations of the current approach are the single fault assumption, and the applicability to only continuous systems. In future work, we plan to extend our approach to multiple faults, based on results presented in [35], and to hybrid systems, based on results presented in [56, 57]. We are also planning to integrate our distributed diagnosis framework into a distributed prognostics framework (that also uses structural model decomposition with PCs [58]) to develop an integrated distributed diagnostics and prognostics framework.

References

[1] P. Struss, Model-based diagnosis for industrial applications, in: Colloquium-Applications of Model-based Reasoning, Institute of Electrical Engineers (IEE), Savoy Place, London, United Kingdom, 1997.

[2] R. Reiter, A Theory of Diagnosis from First Principles, Artificial Intelligence 32 (1987) 57–95.

[3] W. Hamscher, L. Console, J. de Kleer, Readings in Model-based Diagnosis, Morgan-Kaufmann Pub., San Mateo, 1992.

[4] J. J. Gertler, Fault Detection and Diagnosis in Engineering Systems, Marcel Dekker, Inc., New York, NY, 1998.

[5] O. Dressler, P. Struss, The Consistency-based approach to automated diagnosis of devices, in: G. Brewka (Ed.), Principles of Knowledge Representation, CSLI Publications, Standford, 1996, pp. 269–314.

[6] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D. Teneketzis, Diagnosability of discrete-event systems, IEEE Transactions on Automatic Control 40 (9) (1995) 1555–1575.

[7] P. J. Mosterman, G. Biswas, Diagnosis of continuous valued systems in transient operating regions, IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans 29 (6) (1999) 554–565.

[8] I. Roychoudhury, G. Biswas, X. Koutsoukos, Designing distributed diagnosers for complex continuous systems, IEEE Transactions on Automation Science and Engineering 6 (2) (2009) 277–290.

[9] M. J. Daigle, X. Koutsoukos, G. Biswas, A qualitative event-based approach to continuous systems diagnosis, IEEE Transactions on Control Systems Technology 17 (4) (2009) 780–793.

[10] M. Daigle, I. Roychoudhury, G. Biswas, X. Koutsoukos, An Event-based Approach to Distributed Diagnosis of Continuous Systems, in: Proceedings of the 21st International Workshop on Principles of Diagnosis (DX10), Portland, Oregon, USA, 2010, pp. 15–22.

[11] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, Diagnosis and Fault-Tolerant Control, Springer, 2006.

[12] L. Travé-Massuyès, T. Escobet, X. Olive, Diagnosability analysis based on component supported analytical redundancy relations, IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans 36 (6).

[13] M. Krysander, J. Aslund, M. Nyberg, An efficient algorithm for finding minimal over-constrained sub-systems for model-based diagnosis, IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans 38 (1).

[14] B. Pulido, C. Alonso-González, Possible Conflicts: a compilation technique for consistency-based diagnosis, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 34 (5) (2004) 2192–2206.

[15] A. Bregon, G. Biswas, B. Pulido, A Decomposition Method for Nonlinear Parameter Estimation in TRANSCEND, IEEE Trans. Syst. Man. Cy. Part A 42 (3) (2012) 751–763.

[16] E. Balaban, S. Narasimhan, M. Daigle, J. Celaya, I. Roychoudhury, B. Saha, S. Saha, K. Goebel, A Mobile Robot Testbed for Prognostics-Enabled Autonomous Decision Making, in: Annual Conference of the Prognostics and Health Management Society (PHM11), Montreal, Canada, 2011, pp. 15–30.

[17] E. Balaban, S. Narasimhan, M. Daigle, I. Roychoudhury, A. Sweet, C. Bond, G. Gorospe, Development of a Mobile Robot Test Platform and Methods for Validation of Prognostics-Enabled Decision Making Algorithms, International Journal of Prognostics and Health Management 4 (1).

[18] I. Roychoudhury, M. Daigle, A. Bregon, B. Pulido, A structural model decomposition framework for systems health management, in: Proceedings of the 2013 IEEE Aerospace Conference, 2013.

[19] M. Chantler, S. Daus, T. Vikatos, G. Coghill, The use of quantitative dynamic models and dependency recording engines, in: Proc. of the 7th Intl. WS. on Principles of Diagnosis, DX96, Val Morin, Quebec, Canada, 1996, pp. 59–68.

[20] G. Biswas, M. Cordier, J. Lunze, L. Travé-Massuyè, M. Staroswiecki, Diagnosis of complex systems: bridging the methodologies of the FDI and DX communities, IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics 34 (5) (2004) 2159–2162.

[21] J. de Kleer, J. Kurien, Fundamentals of model-based diagnosis, in: Preprints of the 5th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes, SAFEPROCESS03, Washington D.C., USA, 2003.

[22] M. Kinnaert, Fault Diagnosis Based on Analytical Models for Linear and Nonlinear Systems - A Tutorial.

[23] M. Cordier, P. Dague, F. Lévy, J. Montmain, M. Staroswiecki, L. Travé-Massuyès, Conflicts versus Analytical Redundancy Relations: a comparative analysis of the Model-based Diagnosis approach from the Artificial Intelligence and Automatic Control perspectives, IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics 34 (5) (2004) 2163–2177.

[24] V. Brusoni, L. Console, P. Terenziani, D. Theseider-Dupre, A spectrum of definitions for temporal model-based diagnosis, Artificial Intelligente 102 (1) (1998) 39–80.

[25] E. Loiez, P. Taillibert, Polynomial temporal band sequences for analog diagnosis, in: Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97), Nagoya, Japan, 1997, pp. 474–479.

[26] L. Travé-Massuyès, T. Escobet, J. Quevedo, The causal qualitative fault detection and diagnosis system CAEN and its application in the gas turbine domain, in: QMFDI Vacation School, DAMADICS Excellence Network, 2000.

[27] P. Struss, Fundamentals of model-based diagnosis of dynamic systems, in: International Joint Conference on Artificial Intelligence, Vol. 15, 1997, pp. 480–485.

[28] J. de Kleer, B. C. Williams, Diagnosing multiple faults, Artificial Intelligente 32 (1987) 97–130.

[29] J. Armengol, A. Bregon, T. Escobet, E. Gelso, M. Krysander, M. Nyberg, X. Olive, B. Pulido, L. Travé-Massuyès, Minimal Structurally

Overdetermined sets for residual generation: A comparison of alternative approaches, in: Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, SAFEPROCESS09, Barcelona, Spain, 2009, pp. 1480–1485.

[30] B. Pulido, A. Bregon, C. Alonso-Gonzalez, Analyzing the influence of differential constraints in Possible Conflict and ARR computation, in: Current Topics in Artficial Intelligence, CAEPIA 2009 Selected Papers. P. Meseguer, L. Mandow, R. M. Gasca Eds., Springer-Verlag Berlin, 2009.

[31] P. Struss, O. Dressler, Physical negation: Introducing fault models into the General Diagnostic Engine, in: Proceedings of the 11th International Joint Conference on Artifical Intelligence, IJCAI89, Detroit, Michigan, USA, 1989, pp. 1318–1323.

[32] R. Isermann, P. Ballé, Trends in the Application of Model-Based Fault Detection and Diagnosis of Technical Processes, Control Eng. Practice 5 (5) (1997) 709–719.

[33] E.-J. Manders, S. Narasimhan, G. Biswas, P.-J. Mosterman, A combined qualitative/quantitative approach for fault isolation in continuous dynamic systems, in: SafeProcess 2000, Vol. 1, Budapest, Hungary, 2000, pp. 1074–1079.

[34] M. J. Daigle, X. D. Koutsoukos, G. Biswas, Distributed diagnosis in formations of mobile robots, IEEE Transactions on Robotics 23 (2) (2007) 353–369.

[35] M. Daigle, A qualitative event-based approach to fault diagnosis of hybrid systems, Ph.D. thesis, Vanderbilt University (2008).

[36] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D. Teneketzis, Failure diagnosis using discrete-event models, IEEE Transactions on Control Systems Technology 4 (2) (1996) 105–124.

[37] R. Debouk, S. Lafortune, D. Teneketzis, Coordinated decentralized protocols for failure diagnosis of discrete event systems, Discrete Event Dynamic Systems 10 (1-2) (2000) 33–86.

[38] A. G. Mutambara, Decentralized Estimation and Control for Multisensor Systems, CRC Press, Boca Raton, 1998.

[39] A. Samantaray, B. Bouamama, Model-Based Process Supervision: A Bond Graph Approach, Springer Verlag, London, UK, 2008.

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

[41] S. J. Julier, J. K. Uhlmann, A new extension of the Kalman filter to nonlinear systems, in: Proceedings of the 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls, 1997, pp. 182–193.

[42] M. S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, IEEE Transactions on Signal Processing 50 (2) (2002) 174–188.

[43] G. Biswas, G. Simon, N. Mahadevan, S. Narasimhan, J. Ramirez, G. Karsai, A robust method for hybrid diagnosis of complex systems, in: Proceedings of the 5th Symposium on Fault Detection, Supervision and Safety for Technical Processes, 2003, pp. 1125–1131.

[44] I. Roychoudhury, G. Biswas, X. Koutsoukos, Comprehensive Diagnosis of Continuous Systems Using Dynamic Bayes Nets, in: Proceedings of the 19th International Workshop on Principles of Diagnosis (DX08), Blue Mountains, Australia, 2008, pp. 151–158.

[45] S. H. Zad, R. Kwong, W. Wonham, Fault diagnosis in discrete-event systems: framework and model reduction, IEEE Transactions on Automatic Control 48 (7) (2003) 1199–1212.

[46] R. Su, W. M. Wonham, Global and local consistencies in distributed fault diagnosis of discrete-event systems, IEEE Transactions on Automatic Control 15 (12) (2005) 1923–1935.

[47] P. Baroni, G. Lamperti, P. Pogliano, M. Zanella, Diagnosis of large active systems, Artificial Intelligence 110 (1) (1999) 135–183.

[48] J. Kurien, X. Koutsoukos, F. Zhao, Distributed diagnosis of networked embedded systems, in: Proceedings of the 13th International Workshop on Principles of Diagnosis (DX-02), Semmering, Austria, 2002, pp. 179–188.

[49] S. Indra, L. Travé-Massuyès, E. Chanthery, A decentralized FDI scheme for spacecraft: Bridging the gap between model based FDI research and practice, in: 4th European Conference for Aerospace Sciences, St Petersburg, Russia, 2011.

[50] J. Koscielny, K. Zakroczymski, Fault isolation method based on time sequences of symptom appearance, in: Proceedings of IFAC Safaprocess, Budapest, Hungary, 2000.

[51] V. Puig, J. Quevedo, T. Escobet, B. Pulido, On the integration of fault detection and isolation in model-based fault diagnosis, in: Proceedings of the 16th International Workshop on Principles of Diagnosis (DX-05), 2005, pp. 227–232.

[52] V. Puig, F. Schmid, J. Quevedo, B. Pulido, A new fault diagnosis algorithm that improves the integration of fault detection and isolation, in: Proceedings of the 44th IEEE Conference on Decision and Control, 2005, pp. 3809–3814.

[53] M. Bayoudh, L. Travé-Massuyès, X. Olive, Hybrid systems diagnosability by abstracting faulty continuous dynamics, in: Proc. of the 17th Int. Workshop on Principles of Diagnosis, 2006, pp. 9–15.

[54] J. Meseguer, V. Puig, T. Escobet, Fault Diagnosis Using a Timed Discrete-Event Approach Based on Interval Observers: Application to Sewer Networks, IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans 40 (5) (2010) 900–916.

[55] M. Cordier, C. Dousson, Alarm driven monitoring based on chronicles, in: Proceedings of Safeprocess, 2000, pp. 286–291.

[56] M. Daigle, I. Roychoudhury, G. Biswas, X. Koutsoukos, A. Patterson-Hine, S. Poll, A comprehensive diagnosis methodology for complex hybrid systems: A case study on spacecraft power distribution systems, IEEE Transactions of Systems, Man, and Cybernetics, Part A 4 (5) (2010) 917–931.

[57] A. Bregon, C. Alonso, G. Biswas, B. Pulido, N. Moya, Hybrid systems fault diagnosis with possible conflicts, in: Proceedings of the 22nd International Workshop on Principles of Diagnosis, Murnau, Germany, 2011, pp. 195–202.

[58] M. Daigle, A. Bregon, I. Roychoudhury, Distributed Damage Estimation for Prognostics based on Structural Model Decomposition, in: Annual Conference of the Prognostics and Health Management Society (PHM11), Montreal, Canada, 2011, pp. 198–208.