An Integrated Framework for Distributed Diagnosis of Process and Sensor Faults

Anibal Bregon
University of Valladolid
Valladolid, 47011, Spain
anibal@infor.uva.es

Matthew Daigle
NASA Ames Research Center
Moffett Field, CA 94035
matthew.j.daigle@nasa.gov

Indranil Roychoudhury
SGT Inc., NASA Ames Research Center
Moffett Field, CA 94035
indranil.roychoudhury@nasa.gov

Abstract— **Complex engineering systems require efficient online fault diagnosis methodologies to improve safety and reduce maintenance costs. In complex systems, faults may occur in the process itself but also in the sensors monitoring the system, which makes the fault diagnosis task difficult, because the signals from which diagnostic reasoning takes place may be corrupted by faulty sensors. As such, many diagnosis solutions focus on either process or sensor faults, but not both. When considering both types of faults, additional diagnostic information is needed because of the additional ambiguity introduced by potentially faulted sensors. As such, traditional centralized diagnosis approaches, which already do not scale well, scale even worse. To address these issues, this paper presents a distributed diagnosis framework for physical systems applied to diagnosis of both sensor and process faults. Using a structural model decomposition method, we develop a distributed diagnoser design algorithm to build local fault diagnosers. These diagnosers are constructed based on global diagnosability analysis of the system, determining the minimal number of residuals required to have the maximum possible diagnosability in the system. We evaluate the design approach on a diagnostic benchmark system that is functionally representative of a spacecraft electrical power distribution system. Results demonstrate that the proposed distributed approach scales significantly better than a centralized approach.**

TABLE OF CONTENTS

1 INTRODUCTION 1
2 RELATED WORK 2
3 SYSTEM MODELING 2
4 RESIDUAL GENERATION AND QUALITATIVE FAULT ISOLATION 4
5 PROBLEM FORMULATION 5
6 APPROACH 6
7 RESULTS 7
8 CONCLUSIONS 9
ACKNOWLEDGMENTS 9
REFERENCES 9
BIOGRAPHY 10

1. INTRODUCTION

Fault diagnosis, an important aspect of systems health management, is essential for ensuring safe, correct, and efficient operation of complex engineering systems. Fault diagnosis involves fault detection (whether system behavior is off-nominal), fault isolation (what is the root cause of the off-nominal behavior), and fault identification (what is the fault magnitude). Fault diagnosis is carried out by using the information provided the the system sensors, hence, when sensors are potentially faulty, the diagnosis process becomes more complex. Due to this, many diagnostic approaches do not consider the diagnosis problem combining both process and sensor faults. When considering both sets of faults, additional diagnostic information is required in order to resolve the ambiguities that arise from diagnostic reasoning over signals from potentially faulty sensors.

Typically, centralized diagnosis solutions have been proposed for fault diagnosis, however these solutions have several inherent shortcomings, the most notable of which is that they do not scale well as the size of the system grows [1–3]. With the additional diagnostic information needed to distinguish among process and sensor faults, they scale even worse. In addition, if the centralized diagnoser fails, the system will have to operate without a diagnosis system, and so they lack robustness. These problems encourage the development of distributed diagnosis frameworks for complex dynamic systems.

In previous work [4], we proposed an approach for distributed diagnosis of process faults based on the analysis of model-based residuals (a residual being the difference between a measured sensor output and the predicted value of that sensor output). Distributed diagnosis was performed by using local models of the system, which were used to make predictions of measured outputs. Local models were generated by decomposing the global model of the system into minimal submodels using structural model decomposition [5]. In [4], we only considered process faults, however, since the decomposition is done by using sensors of the global model as local inputs for the submodels, the performance of the fault diagnosis system depends, among other things, on the sensors from which diagnostic information can be extracted. Consequently, if a sensor fault occurs, incorrect information will be sent to the diagnoser, which will generate incorrect diagnostic conclusions if sensors are not considered to be potentially faulty. Further, in practice, sensors are usually the most common components to fail in a system. Therefore, the approach must be extended to incorporate diagnosis of sensor faults.

The main idea of diagnosis based on structural model decomposition is that by using sensor signals as inputs to the local submodels, faults become decoupled from residuals, thus increasing the diagnosability of the system. Moreover, if some residuals share common variables, linear combinations of these residuals can produce new independent residuals. As it has been demonstrated in the literature (e.g. in [6]), these new residuals do not provide additional isolation capabilities to the diagnosis system for process faults, but they can increase diagnosability for sensor faults. Therefore, the problem of distributed diagnosis including process and sensor faults is directly related to one of residual generation and residual selection to obtain the maximum possible diagnosability in the system. However, the total number of possible residual combinations is exceedingly large and most of them provide

978-1-4799-5380-6/15/$31.00

redundant diagnosability information.

In this paper, we formulate a distributed fault diagnoser design problem for process and sensor faults and establish its search space. We propose new definitions for global diagnosability, extended from those presented in [4], which form the theoretical basis for designing distributed local diagnosers with guarantees that these local diagnosers can be run independently, do not need a central coordinator, and provide the same overall diagnosability as the centralized diagnoser. We provide a diagnoser design algorithm that searches in the residual space to find optimal global and local diagnoser designs. A greedy variant of the algorithm can find locally optimal solutions with minimal computational effort. We show that finding a distributed diagnoser design has significantly better design-time requirements than finding a centralized diagnoser design. Finally, we demonstrate the local distributed diagnosers design approach on the Advanced Diagnostics and Prognostics Testbed (ADAPT) [7], an electrical power system testbed that has served as a benchmark diagnostic system in the diagnostics community [8, 9].

This paper is organized as follows. Section 2 presents related work to set the context for our contributions. Section 3 provides background information on our system modeling and structural model decomposition approaches. Section 4 presents our qualitative fault isolation framework. Section 5 presents the problem formulation. Section 6 describes the distributed diagnoser design approach, and Section 7 provides the ADAPT case study and diagnoser design results. Finally, Section 8 concludes the paper and discusses future work.

2. Related Work

Different approaches for sensor fault diagnosis have been presented in the scientific literature [10–14]. Typically, in a large majority of these approaches, residuals are designed such that they are sensitive to sensor and other component faults. Then, analysis of these residuals is carried out to solve the diagnosis problem. The differences between the approaches mainly include how residuals are constructed from the system model, and what approach is adopted for the observers that generate estimates of expected outputs. Some approaches for pure sensor fault diagnosis include the approach of [10], in which the authors present a state and noise estimator for *descriptor systems* that completely decouple both the input and output disturbances; the approach of [11] that uses a novel Linear Matrix Inequality (LMI)-based observer design for Lipschitz nonlinear systems for sensor fault diagnosis; and the approach of [12], in which a principle component analysis (PCA) method for detecting and isolating sensor faults in representative air-handling units is presented. In all of these approaches, if process faults occur, the sensor fault diagnosis systems will produce incorrect results. Likewise, approaches that ignore sensor faults and consider only process faults produce incorrect diagnoses when sensor faults occur. Some diagnosis approaches consider both sets of faults, including the approach of [13], in which optimization techniques are used for diagnosis of gas turbine engine components and sensor faults; and the approach of [14], in which a Radial Basis Function (RBF) neural network is used for sensor and component fault diagnosis in chemical processes. Similarly, our approach tackles both sensor and process faults, but, unlike these approaches, is based on structural analysis of the system models. Further, our approach is distributed and addresses many of the drawbacks of typical centralized diagnosis approaches, such as a single point of failure, high computational complexity, and poor scalability.

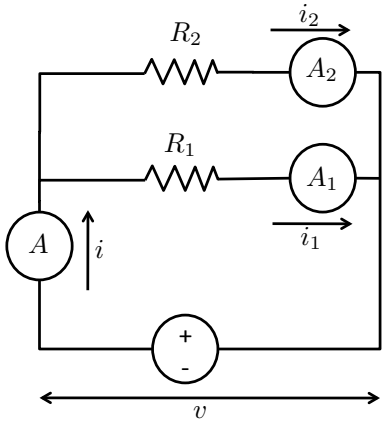

Figure 1. Electrical circuit running example.

As mentioned in the previous section, the diagnoser design problem in our approach becomes one of residual selection, which is related to the problem of sensor placement. Our work is in contrast to other approaches present in literature [15–17] in that we look for solutions that obtain maximum diagnosability by minimizing the size of the submodels, which yields smaller-sized diagnosers and allows its implementation as a distributed approach. For example, in [15], the authors propose an approach for optimal sensor location to increase the fault detection performance in dynamic systems using statistical tests. In [16] the authors assume that the system is diagnosable given a set of sensors and look for the least expensive combination of those sensors under which the system is still diagnosable.

3. System Modeling

In this section, we first describe our approach to system modeling. We then describe our structural model decomposition approach, which, given a global system model, creates local models of system behavior. We adopt here the structural model decomposition framework described in [5]. In the following, we review the main details and refer the interested reader to [5] for additional explanation.

System Model

We define a model as follows:

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 five disjoint sets, namely, the set of state variables, X; the set of parameters, Θ; the set of inputs, U; the set of outputs, Y; and the set of auxiliary variables, A. Each constraint $c = (\varepsilon_c, V_c)$, such that $c \in C$, consists of an equation ε_c involving variables $V_c \subseteq V$.

Input variables, U, are known, and 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. Auxiliary variables, A, are additional variables that are algebraically related to the state and parameter variables, and are used to simplify the structure of the equations.

Example 1: Throughout the paper, we will use a simple

electrical circuit as a running example, shown in Fig. 1. The system consists of a voltage source and two resistances, R_1 and R_2, connected in parallel. Several sensors measure the current in each branch (i, i_1, and i_2) and the total voltage (v). The model $\mathcal{M}^*$ is represented by the variable sets $X = \varnothing$, $\Theta = \{R_1, R_2, i_1^b, i_2^b, i^b, v^b\}$, $U = \{S_v\}$, $Y = \{i_1^*, i_2^*, i^*, v^*\}$, and $A = \{i_1, i_2, i, v\}$; and the set of constraints $C = \{c_1, c_2, \ldots, c_8\}$, where the constraints are given as follows:

$$i = i_1 + i_2, \quad (c_1)$$
$$i_1 = \frac{v}{R_1}, \quad (c_2)$$
$$i_2 = \frac{v}{R_2}, \quad (c_3)$$
$$v = S_v, \quad (c_4)$$
$$i^* = i + i^b, \quad (c_5)$$
$$i_1^* = i_1 + i_1^b, \quad (c_6)$$
$$i_2^* = i_2 + i_2^b, \quad (c_7)$$
$$v^* = v + v_b. \quad (c_8)$$

Here, the asterisk superscript is used to denote a measured value of a physical variable, e.g., i_1 is the current and i_1^* is the measured current. Since i_1 is used to compute other variables, like i, it cannot belong to Y and a separation of the variables is required. Also, the b superscript is used to indicate a sensor bias parameter.

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 (*i*) for all $v \in U \cup \Theta$, $\mathcal{A}$ does not contain any α such that $\alpha = (c, v)$, i.e., input or parameter variables cannot be the dependent variables in the causal assignment); (*ii*) for all $v \in Y$, $\mathcal{A}$ does not contain any $\alpha = (c, v_c^{out})$ where $v \in V_c - \{v_c^{out}\}$, i.e., a measured variable can be used as the dependent variable; and (*iii*) for all $v \in V - U - \Theta$, $\mathcal{A}$ contains exactly one $\alpha = (c, v)$, i.e., every variable that is not input or parameter is computed by only 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: The causal model $\mathcal{M}$ for the running example is represented by the same variables and constraints as $\mathcal{M}^*$, along with the set of causal assignments $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_8\}$, as given below:

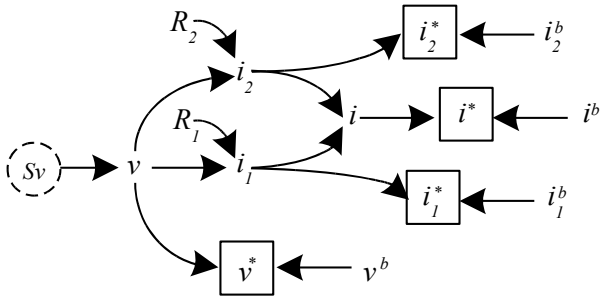

Figure 2. Causal graph of the three-tank system.

$$i := i_1 + i_2, \quad (\alpha_1)$$
$$i_1 := \frac{v}{R_1}, \quad (\alpha_2)$$
$$i_2 := \frac{v}{R_2}, \quad (\alpha_3)$$
$$v := S_v, \quad (\alpha_4)$$
$$i^* := i + i_b, \quad (\alpha_5)$$
$$i_1^* := i_1 + i_1^b, \quad (\alpha_6)$$
$$i_2^* := i_2 + i_2^b, \quad (\alpha_7)$$
$$v^* := v + v_b. \quad (\alpha_8)$$

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 electrical system running example is given in Fig. 2. In the graph, we mark inputs with dashed circles and outputs with solid squares.

Structural Model Decomposition

In our approach, a fault f is modeled as a step change in a system model parameter value, $\theta \in \Theta$. Faults cause changes in observed system behavior from model-predicted behavior. We can detect such changes by computing residuals, defined as the difference between the measured and predicted value of some sensor. Using the causal model $\mathcal{M}$ of a system, we can predict values of all the sensors in order to compute residuals. However, in the global model, faults are typically coupled to all the sensors, i.e., they cause deviations in all the sensors eventually. Through structural model decomposition, we can instead define local submodels for the purpose of computing residuals, in which each residual responds to only a subset of the faults, increasing diagnosability [18].

Under this approach, given a (global) model, we can create (local) submodels that use as additional inputs values from the sensors [5]. 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), 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 5 (Causal Submodel) A *causal submodel* $\mathcal{M}_{Y_i}$ of

a causal model $\mathcal{M} = (V, C, \mathcal{A})$ is a tuple $\mathcal{M}_{Y_i} = (V_i, C_i, \mathcal{A}_i)$, where $V_i \subseteq V$, $C_i \subseteq C$, and $\mathcal{A}_i \cap \mathcal{A} \neq \varnothing$.

When using measurements (from Y) as local inputs for a causal submodel, the causality of these constraints must be reversed, and so, in general, $\mathcal{A}_i$ is not a subset of $\mathcal{A}$.

The procedure for generating a submodel from a causal model is given as Algorithm 1 (`GenerateSubmodel`) in [5]. Given a causal model $\mathcal{M}$, a set of variables that are considered as local inputs, U^*, and a set of variables to be computed, V^*, the `GenerateSubmodel` algorithm derives a causal submodel $\mathcal{M}_i$ that computes V^* using U^*. The algorithm works by starting at the variables in V^*, and propagating backwards through the causal dependencies. Propagation along a dependency chain stops once a variable in U^* is reached, or once a constraint is reached in which the causality can be reversed so that a variable in U^* can become a local input. We refer the reader to [5] for the algorithm pseudocode and additional details.

4. Residual Generation and Qualitative Fault Isolation

As mentioned in Section 1, the goal of this work is to solve a distributed diagnosis problem with sensor and process faults such that the maximum diagnosability is achieved. The solution of this problem depends on the diagnosis framework chosen. In this section, we briefly present the fundamentals of our fault isolation approach. For details, please refer to [4, 19, 20].

As previously mentioned, in our approach, a fault f is modeled as a step change in a system model parameter value, $\theta \in \Theta$. 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. The complete fault set is denoted as F.

Example 4: In the electrical system running example (Fig. 1), the complete fault set F consists of $\{R_1^-, R_1^+, R_2^-, R_2^+, i^{b-}, i^{b+}, i_1^{b-}, i_1^{b+}, i_2^{b-}, i_2^{b+}, v^{b-}, v^{b+}\}$.

Faults cause transients in the system variables that are observed as deviations of measured values from predicted values. This is captured through the concept of a residual.

Definition 6 (Residual) A *residual*, r_y, is a time-varying signal that is computed as the difference between a measurement, $y \subseteq Y$, and a predicted value of the measurement y, denoted as $\hat{y}$. A set of residuals is denoted as R.

From Section 3, we see that there are several potential submodels that can compute $\hat{y}$, depending on what local inputs are available. For a given y, we can find all submodels computing a version of $\hat{y}$ (and thus defining a residual) using the model decomposition algorithm. To compute y for a given submodel, we can use as local inputs U and measured sensor signals from $Y - \{y\}$ to define U^* for the `GenerateSubmodel` algorithm. For a given U^*, only a subset of U^* may actually end up as local inputs to the submodel, so different calls to `GenerateSubmodel` may yield the same submodel to compute y. We define the complete residual set as follows.

Definition 7 (Complete Residual Set) For model $\mathcal{M}$ with inputs U and outputs Y, the *complete residual set* is the set of all residuals, for each $y \in Y$, from submodels $\mathcal{M}_i$ computed from $\mathcal{M}$ using $U^* = U \cup Y^*$ and $V^* = \{y\}$, where $Y^* \in 2^{Y-\{y\}}$.

Informally, the complete residual set contains, for a given set of sensors, every possible way of computing residuals for those sensors. It is denoted as R_Y.

In the nominal situation all residuals are ideally zero, and when a fault occurs they become nonzero. It is through analysis of the residual signals that fault isolation is performed. The transients produced in the residuals are captured as qualitative *fault signatures* [19].

Definition 8 (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 $\Sigma_{f,r}$.

The symbols s_1 and s_2 are selected from $\{0, +, -\}$, denoting no change, increase, and decrease, respectively.

As additional diagnostic information we use also the temporal order of residual deviation, captured through the concept of *relative residual orderings* [21].

Definition 9 (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 $\Omega_{f,R}$.

In order to generate signatures and orderings from a model, we extend the definition of a model to include qualitative labels on causal constraints. For each independent variable involved in a constraint, we associate a qualitative label indicating the qualitiative direction of influence the independent variable has on the dependent variable. A dt label indicates an integration, a + label indicates that a directly proportional change, and a – label indicates an inversely proportional change. From this representation a Temporal Causal Graph [19] (TCG) is obtained, and the algorithms described in [22] may be used to automatically derive the signatures and orderings.[2]

Example 5: The fault signatures and selected relative residual orderings on the global residuals for the electrical systems are shown in Table 1.

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. Each of these possibilities is a *fault trace*.

Definition 10 (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 $\Omega_{f,R}$.

[2] TCGs may also be derived directly from bond graphs [23]. Our modeling approach is more general in that it is not restricted to the system topologies imposed by bond graphs.

Table 1. Fault Signatures and Residual Orderings for the Electrical Circuit

Fault	r_{i^*}	$r_{i_1^*}$	$r_{i_2^*}$	r_{V^*}	Residual Orderings
R_1^+	−	−	0	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
R_1^-	+	+	0	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
R_2^+	−	0	−	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
R_2^-	+	0	+	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
i^{b+}	+	0	0	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
i^{b-}	−	0	0	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
i_1^{b+}	0	+	0	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
i_1^{b-}	0	−	0	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
i_2^{b+}	0	0	+	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
i_2^{b-}	0	0	−	0	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
v^{b+}	0	0	0	+	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$
v^{b-}	0	0	0	−	$r_{i^*} \prec r_{i_2^*}, r_{i^*} \prec r_{v^*}, \ldots$

The set of all fault traces for a fault constitutes its *fault language*.

Definition 11 (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.

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

Definition 12 (Distinguishability) Given a residual set, R, a fault f_i is *distinguishable* from a fault f_j, denoted by $f_i \nsim_R f_j$, 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[3] (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.

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 13 (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.

The diagnosability for a diagnosis model is then defined based on the distinguishability of faults.[4]

Definition 14 (Diagnosability) The *diagnosability* of a diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$, $D_{F,R}$, is the number of fault pairs $(f_i, f_j \in F)\ f_i \neq f_j$ where $f_i \nsim_R f_j$.

It is important to consider here that distinguishability is not a symmetric property, hence we can have situations where $f_i \nsim_R f_j$ but $f_j \sim_R f_i$. In this case, the diagnosability definition will count the pair (f_i, f_j) but not the pair (f_j, f_i). In the best of the cases, diagnosability of a diagnosis model will be equal to $|F| * (|F| - 1)$, and we say that the system has *complete diagnosability*. On the other hand, the worst possible diagnosability is 0.

In order to perform distributed fault diagnosis, 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 global fault set, F. Specifically, the fault set F is partitioned (either manually or by an automatic method) into local fault sets, such that every fault in F is included in exactly one local fault set, i.e., each local diagnoser is responsible for correctly isolating the faults in its local fault set.

Definition 15 (Diagnosis Submodel) A *diagnosis submodel* $\mathcal{S}_i$ of 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 $R_i \subseteq R$.

The global correctness condition [4] is that if a fault occurs the diagnoser responsible for it should identify it, and all other diagnosers should not identify any of their local faults as having occurred. However, 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 (which are distinguishable for $\mathcal{S}$) 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 correct in a global context. Therefore, we require an extended notion of diagnosability, called *global diagnosability*, that takes into account these concepts.

Definition 16 (Global Diagnosability) The *global diagnosability* of a diagnosis submodel $\mathcal{S}_i = (F_i, Y_i, R_i, L_{F_i,R_i})$ from diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$, $D^F_{F_i,R_i}$, is the number of fault pairs $(f_i \in F_i, f_j \in F)\ f_i \neq f_j$ where $f_i \nsim_{R_i} f_j$.

The best possible global diagnosability is then $|F_i|(|F| - 1)$, and the worst is 0.[5]

For distributed diagnosis, the design problem will be one of residual assignment to diagnosis submodels.[6] As we will describe in the following section, solutions to this problem are only ones in which we obtain the best possible global diagnosability. If we partition the global fault set F into the local fault sets for the diagnosis submodels, then all faults will be covered. If we then design (i.e., assign residuals to) each diagnosis submodel such that they all have the best possible global diagnosability, then the local diagnosers will generate results that are globally correct, i.e., the result will be the same as that produced by a centralized diagnoser with the best possible diagnosability.

5. PROBLEM FORMULATION

The problem we are trying to solve is to design distributed diagnosers for a given system. The design problem is to select an optimal set of residuals for each local diagnoser in order to

[3] 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 $\lambda_i \lambda_k = \lambda_j$.

[4] The diagnosability definitions presented here are different from our earlier work, in [4], in that they convey the amount of diagnosability of a system, i.e., diagnosability is a number. In previous work, diagnosability was a Boolean indicating whether a diagnosis model was completely diagnosable or not.

[5] Diagnosability is a special case of global diagnosability in which there is only one diagnosis submodel equal to $\mathcal{S}$.

[6] For centralized diagnosis, the design problem is one of residual assignment to a diagnosis model. This problem is a special case of the distributed diagnoser design problem where there is only one diagnosis model.

achieve the maximum possible diagnosability covering both process and sensor faults. We define the *maximum diagnosability* as the highest possible value of diagnosability for a given system with model $\mathcal{M}$. This maximum diagnosability is always obtained when the complete residual set, R_Y, is used. However, the maximum diagnosability can often be achieved with only a subset of R_Y, since many residuals provide redundant information given other residuals, and thus do not improve diagnosability.

For a given y, there are a total of $2^{|Y|-1}$ possible subsets of $Y - \{y\}$ from which to define U^*, and so there are at most that many unique ways to compute a residual for y. For $|Y|$ sensors then, there are a total of $|Y|2^{|Y|-1}$ potential residuals to choose from for each local diagnoser, i.e., R_Y is at most that large. For a given local diagnoser, a candidate solution is then a set of residuals $R_i \subseteq R_Y$. There are at most $2^{|Y|2^{|Y|-1}} - 1$ candidate solutions. A candidate solution is a solution if it achieves the maximum diagnosability, i.e., the same diagnosability as with R_Y.

An optimal solution is one that satisfies some given criteria the best, such as using the minimum number of residuals. We define a relational operator $\prec$ for solutions, describing which solutions are preferred over others and thus obtaining a notion of optimality for solutions. The $\prec$ operator depends on the particular application, and we will describe an implementation of it in the following section.

The problem can then be formally defined as follows.

Problem 1: For a model $\mathcal{M}$ and local fault sets F_i, the problem is to find, for each F_i, a set of residuals R_i such that there is no $R_j \neq R_i$ where $R_j \prec R_i$.

6. APPROACH

As discussed in Section 3, for a given system we define a model $\mathcal{M}$ and a set of faults F. From this, we can generate the complete residual set R_Y and the set of fault languages L_{F,R_Y} from which we can define its diagnosability. As discussed in Section 5, a solution to the residual selection problem for a local fault set F_i is a set of residuals $R_i \subseteq R_Y$ that has the maximum attainable diagnosability. The way in which we should most efficiently search the solution space to find the optimal set of solutions depends on the particular implementation of the $\prec$ operator for candidate solutions [24].

What makes one solution better than another can be defined using several metrics. First and foremost, one solution is better than another if it has better diagnosability, i.e., it has fewer pairs of indistinguishable faults for the diagnostic information available from its residuals. We also prefer solutions with a smaller number of residuals, as these solutions will be simpler to implement and more efficient. Further, we prefer solutions with more analytical redundancy, which can be measured as the ratio of the number of residuals to the number of input sets that compute them. The more diversity in the residual inputs, the more robust the solution will be.

So, as an implementation of $\prec$, we adopt the following for local fault set F_i: (*i*) $R_i \prec R_j$ if $D_{F_i,R_i} > D_{F_i,R_j}$; (*ii*) if $D_{F_i,R_i} = D_{F_i,R_j}$, $R_i \prec R_j$ if $|R_i| < |R_j|$; (*iii*) if $D_{F_i,R_i} = D_{F_i,R_j}$ and $|R_i| = |R_j|$, $R_i \prec R_j$ if $|R_i|/|\mathbb{U}(R_i)| < |R_j|/|\mathbb{U}(R_j)|$, where, for a given R, $\mathbb{U}(R)$ is defined as the set of input sets used to compute the residuals, i.e., $\mathbb{U}(R) = \{U_i : \forall r_i \in R, U_i \in \mathcal{M}_{r_i}\}$ and $\mathcal{M}_{r_i}$ refers to the submodel computing r_i.

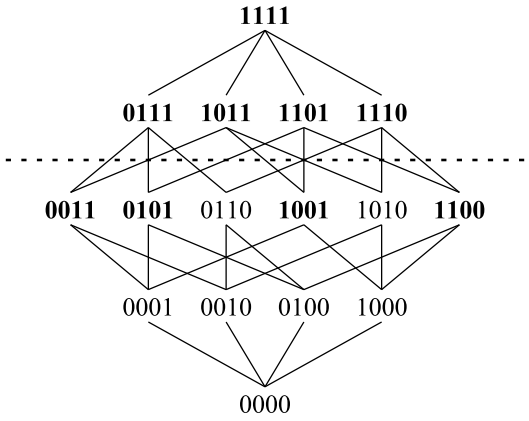

Figure 3. Solution lattice.

There are many ways to search the candidate solution space to find the set of optimal solutions. One may picture the search space as a lattice, as shown in Fig. 3 for a search space of 4 residuals, where a 1 represents inclusion of a residual $r_i \in R_Y$ and a 0 indicates exclusion of a residual. Moving from one solution to another along an illustrated line indicates adding a residual (moving up in the lattice) or removing a residual (moving down in the lattice). Within the candidate solution space, a subset of the candidates are solutions, and a subset of these are optimal. In Fig. 3, candidate solutions are denoted in boldface type. Note that if one residual set R is a solution, then any residual set $R' \supseteq R$ is also a solution (i.e., adding residuals cannot decrease diagnosability). We prefer solutions that have the minimum number of residuals, so assuming all else is equal, the solutions that are below the dashed line in Fig. 3 would constitute the set of optimal solutions.

We need to search this space in the most efficient way. A typical search strategy would be to start at some initial solution, and try to improve it by moving in the lattice. For a bottom-up search, we start at the bottom of the lattice, i.e., the solution without any residuals, and move up in the lattice, adding residuals, until we obtain solutions with the maximum diagnosability. For a top-down search, we start at the top of the lattice, i.e., the solution with all residuals, and move down in the lattice, removing residuals, until we begin to lose diagnosability. Either approach is valid, however, a bottom-up search is likely to be more efficient, because it is much more likely, from our experience, that optimal solutions lie in the bottom half of the lattice, i.e., less than half the residuals in R_Y are required. Since we prefer solutions with fewer residuals, it makes additional sense to use a bottom-up search strategy in a breadth-first manner, where we move up layer-by-layer in the lattice to find a set of minimum-residual solutions. We can then reduce this set to the set of optimal solutions.

The bottom-up search algorithm is given as Algorithm 1. The algorithm maintains a solution set $\mathcal{R}^*$, a queue of solutions to search next, $\mathcal{R}$, and a set of solutions that have been tried, $\mathcal{R}_{\text{tried}}$. The search starts at the solution with no residuals, $\varnothing$. It goes through the solution queue, popping the next off the queue, then examining all child solutions containing one more residual that have not been examined yet. If the new

Algorithm 1 $\mathcal{R}^* = \texttt{BottomUpSearch}(F_i, R)$

```
1:  R* ← ∅
2:  R ← ∅
3:  R_tried ← ∅
4:  n_R ← ∞
5:  while R ≠ ∅ do
6:     R' ← pop(R)
7:     if |R'| < n_R then
8:        for all r ∈ R_Y − R do
9:           R'' ← R ∪ {r}
10:          if R'' ∉ R_tried then
11:             R_tried ← R_tried ∪ R''
12:             if R'' ≺ R' then
13:                R* ← R* ∪ {R''}
14:                if D^F_{F_i,R''} < D^F_{F_i,R} then
15:                   R ← push(R, {R''})
16:                else
17:                   n_R ← min(n_R, |R''|)
18:                end if
19:             end if
20:          end if
21:       end for
22:    end if
23: end while
24: R ← R*
25: R* ← {R(1)}
26: for all R ∈ R do
27:    if R ≺ R*(1) then
28:       R* ← {R}
29:    else if R*(1) ⊀ R then
30:       R* ← R* ∪ {R}
31:    end if
32: end for
```

solution is better, it is added to the solution set, and to the queue. Otherwise, the search along this path will terminate, because it is not possible to improve the solution further. Once we find a solution with the maximum diagnosability at the current breadth, stored in n_R, we can stop the search once we finish examining all solutions in the queue at that breadth, because at that point adding a residual will always result in a less preferable solution. Because the same solution can be reached by multiple paths, the algorithm does not explore solutions that have already been added to $\mathcal{R}$. After the while loop, we prune the solution set to optimal solution set (lines 25-32).

The bottom-up search is optimal, because it searches all solutions at a given breadth, except for those that did not improve its parent solution (if residuals from that solution are part of an optimal solution, they will be covered by other search paths). A greedy version of the algorithms can also be specified, where at each breadth, we branch from only the n best solutions. By reducing the branching factor efficiency is increased substantially, but at the cost of potentially missing optimal solutions. However, we are still guaranteed that the solutions have the best attainable diagnosability, since, in the worst case, it reaches the top of the lattice, i.e., the solution with all residuals.

For a centralized diagnoser, we call $\texttt{BottomUpSearch}(F, R_Y)$. Since the candidate solution space is exponential in the number of residuals, even the bottom-up breadth-first search can quickly become expensive, thus motivating the use of greedy algorithms. The number of candidates to search at a given breadth b is at most $\binom{|R_Y|}{b}$. So, the fewer residuals that are needed for an optimal solution, the more efficient the search will be. By obtaining a distributed diagnoser design, in which each diagnoser is assigned a local fault set $F_i \subseteq F$, we can improve design-time efficiency in two different ways. First, the candidate solution space can be smaller, because typically only a subset of R_Y, R_{F_i} responds to the faults in F_i. So we need only search over combinations of residuals in this subset, because residuals that do not respond to any faults in F_i cannot improve diagnosability so need not be considered. Specifically, $2^{|R_{F_i}|} << 2^{|R_Y|}$. Second, the smaller F_i, the smaller the number of residuals needed in a solution, so the needed breadth in the search space is smaller, also reducing computation. For a distributed diagnoser then, we call $\texttt{BottomUpSearch}(F_i, R_{F_i})$.

To take maximum advantage of these properties, we can define F_i in a systematic way, where for each fault parameter θ, we have a local diagnoser and local fault set consisting of $\{\theta^+, \theta^-\}$. For each F_i, we then run $\texttt{BottomUpSearch}(F_i, R_{F_i})$ (or its greedy variant) to obtain solutions.

Example 6: Consider the circuit example. In this case, there are 16 residuals forming R_Y. Thus, there are $2^{16} = 65536$ candidate solutions. For a centralized diagnoser, the bottom up algorithm searches 2514 candidates, and finds 5 optimal solutions, each with perfect diagnosability, 4 residuals, and 1 residual per input set (i.e., the residuals are computed with completely different sets of inputs). In one solution, we have two different residuals for i^*, one for i_1^*, and one for v^*. Other solutions have residuals for different sensors. Using the greedy algorithm, one solution is found that happens to be included in the optimal set of solutions found by the bottom up algorithm, yet it searches only 59 candidates.

Example 7: Consider again the circuit example, but for distributed diagnoser design. Here, we have 6 local fault sets defined covering the 12 faults. The bottom up algorithm ends up searching 224 candidates total over all local fault sets, which is an order of magnitude fewer than in the centralized case. For a given local fault set at most 10 residuals have to be searched, so a maximum of 1024 candidates. At most 59 candidates are searched for each local fault set, and as few as 9. For example, for the local fault set including the R_1 faults, only two residuals are needed with only 1 residual per input set. Other fault sets require only 1 or 2 residuals, significantly reducing the search space compared to the centralized case. The greedy algorithm also finds a subset of the optimal solutions, while searching over 95 candidates total.

7. RESULTS

In this paper, we apply our new methodology to the Advanced Diagnostics and Prognostics Testbed (ADAPT), an electrical power distribution system that is representative of those on spacecrafts. ADAPT serves as a testbed through which faults can be injected to evaluate diagnostic algorithms [7]. ADAPT has been established as a diagnostic benchmark system through the industrial track of the International Diagnostic Competition (DXC) [8, 9, 25]. In particular, this paper is focused on diagnosing faults on a subset of ADAPT, called ADAPT-Lite.

A system schematic for ADAPT-Lite is given in Fig. 4. A battery (BAT2) supplies electrical power to several loads, transmitted through several circuit breakers (CB236, CB262, CB266, and CB280) and relays (EY244, EY260, EY281, EY272, and EY275), and an inverter (INV2) that converts

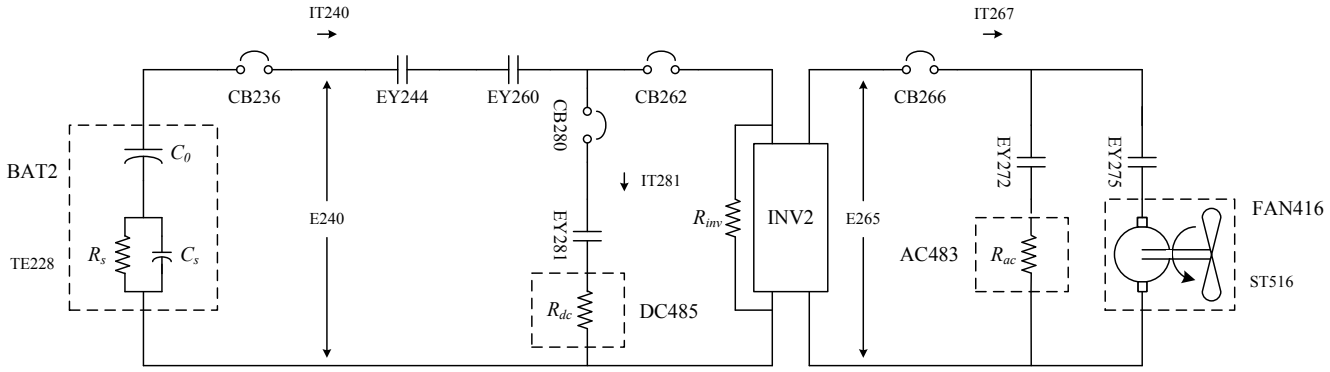

Figure 4. ADAPT-Lite schematic.

dc to ac power. ADAPT-Lite has one dc load (DC485) and two ac loads (AC483 and FAN416). There are sensors throughout the system to report electrical voltage (names beginning with "E"), electrical current ("IT"), one sensor to report the operating state of a load (fan speed, "ST"), and another to report the battery temperature ("TE"). Models and additional details for ADAPT-Lite can be found in [26, 27].

The list of potential faults includes failures in the inverter (INV2), fan (FAN416), DC load (DC485), and AC load (AC483), and faults in different current sensors (IT240, IT267, IT281), voltage sensors (E240, E265), fan speed sensor (ST516), and battery temperature (TE228). For each faulty component, there are two fault modes, one in which there is an increase in the fault parameter and one in which there is a decrease, resulting in a total of 22 faults.

With the available 7 sensors, there are 34 residuals that can be generated to form R_Y. The fault signatures for the residuals computed from the global model inputs are shown in Table 2. For brevity, residual orderings are omitted. Note that with the shown set of residuals, most sensor faults are not distinguishable. For example, if IT267$^+$ occurs, we have to wait infinitely long for other residuals to deviate in order to eliminate INV2$^+$ from consideration. This highlights again that, when also considering sensor faults, we typically require more than one residual per sensor in order to achieve the best diagnosability. With the complete residual set, R_Y, sensor faults become distinguishable, however the system is not completely diagnosable, as the AC load faults are not distinguishable from the fan faults. If an AC load fault occurs, we have to wait, in theory, infinitely long to verify that the fan speed does not deviate. So, an optimal solution will still leave those faults indistinguishable.

First, let us consider the centralized diagnoser case. Having 34 residuals results in a search space of 2^{34}, or about 17 billion. Obviously, an exhaustive search is not an option. From a greedy search, which covers 218 candidates, we find that at least 7 residuals are needed to obtain maximum diagnosability, so around 7 million candidates would need to be searched by the bottom-up search algorithm. The solution found by the greedy algorithm ends up with one residual for E240, four different residuals for IT240 (i.e., computed with four different input sets), one for ST516, and one for TE228. Over the inputs and outputs, it uses all 7 sensors and has six unique input sets, resulting in an average of 1.17 residuals per input set.

Instead, let us consider a distributed diagnoser design. In order to demonstrate the advantage that our proposed definition of the local fault sets gives, we start first with user-defined local fault sets. We define 5 local fault sets with $F_1 = \{\text{TE228}^+, \text{TE228}^-\}$, $F_2 = \{\text{E240}^+, \text{E240}^-, \text{IT240}^+, \text{IT240}^-\}$, $F_3 = \{\text{DC485}^+, \text{DC485}^-, \text{IT281}^+, \text{IT281}^-\}$, $F_4 = \{\text{E265}^+, \text{E265}^-, \text{INV2}^+, \text{INV2}^-, \text{IT267}^+, \text{IT267}^-\}$, and $F_5 = \{\text{AC483}^+, \text{AC483}^-, \text{FAN416}^+, \text{FAN416}^-, \text{ST516}^+, \text{ST516}^-\}$. We find that over all fault sets, the total number of candidates searched by the greedy algorithm is 308. It is larger than for the centralized case because the design algorithm is run 5 times. The bottom-up algorithm, in total, searches 4818 candidates, which is significantly less than 7 million. The optimal solutions need 1, 1, 1, 3, and 3 residuals, respectively. In comparison, the greedy approach also finds solutions with the same minimum number of residuals, and the same number of residuals per input set. It is likely that the smaller the search space, the more likely it is for the greedy algorithm to find optimal solutions. So, the greedy variant may be useful in a distributed design to quickly find good solutions.

Example 8: As an example, a distributed solution found in this case is as follows. For F_1, a single residual for TE228 is used. For F_2, a single residual for E240 is used, which uses IT240 as a local input. For F_3, only a single residual is needed, and using IT281 with IT267 as a local input works. For F_4, three residuals are used: E240 using IT267 as a local input, E265 with no sensors as inputs, and IT240 using E240, E265, IT267, and IT281 as local inputs. For F_5, three residuals are used: E240 with no sensors as local inputs, IT240 with E240, E265, and IT281 as local inputs, and ST516 with E265 as a local input.

Next, we consider a distributed diagnoser design in which we define the local fault sets as described in the previous section. Since the fan and AC load faults are not distinguishable, we group these into the same local fault set, and so have a total of 10 local fault sets. Here, the greedy algorithm searches a total of 297 candidates over all local fault sets, and the bottom-up algorithm searches over 1117 candidates, only a quarter of the number searched with the other local fault set definition. Again, the solutions found by the greedy algorithm are just as good as for the bottom-up algorithm; the only disadvantage is that some optimal solutions are not found using the greedy algorithm.

Example 9: As an example, a distributed solution found in this case is as follows. For the TE228 faults, a single residual for TE228 is needed. For the E240 faults, a single residual is needed: E240 with E265 and IT267 as local inputs. For the

Table 2. Selected Fault Signatures for ADAPT-Lite.

Fault	$E240$	$E265$	$IT240$	$IT267$	$IT281$	$ST516$	$TE228$
FAN416$^+$	0+	00	−⋆	−0	0+	0−	00
AC483$^+$	0+	00	−⋆	−0	0+	00	00
DC485$^+$	0+	00	−⋆	00	−+	00	00
INV2$^+$	0−	+0	+⋆	+0	0−	0+	00
E240$^+$	+0	00	00	00	00	00	00
E265$^+$	00	+0	00	00	00	00	00
IT240$^+$	00	00	+0	00	00	00	00
IT267$^+$	00	00	00	+0	00	00	00
IT281$^+$	00	00	00	00	+0	00	00
ST516$^+$	00	00	00	00	00	+0	00
TE228$^+$	00	00	00	00	00	00	+0

IT240 faults, a single residual is needed: IT281 with IT240 as a local input. For the DC485 faults, a single residual is needed: IT281 with E265 and IT267 as local inputs. For the IT281 faults, a single residual is needed: IT281 with E265 and IT267 as local inputs. For E265 faults, two residuals are needed: IT281 with E265 and IT267 as local inputs, and ST516 with E265 as a local input. For the INV2 faults, two residuals are needed: IT281 with IT267 as an input, and ST516 with no local inputs. For the IT267 faults, two residuals are needed: IT281 with IT267 as a local input, and IT281 with E265 and IT267 as local inputs. For the AC483 and FAN416 faults, three residuals are needed, IT210 with E240, E265, and IT281 as local inputs, IT281 with no local inputs, and ST516 with E265 as a local input. For the ST516 faults, one residual is needed: ST516 with E265 as a local input.

8. Conclusions

In this work, we have presented a diagnosability-based distributed diagnosis solution for sensor and process faults. The solution proposed in this paper analyzes first the diagnosability of the system to determine the maximum attainable diagnosability for single faults in the system. Then, using a structural model decomposition method, we develop a distributed diagnoser design algorithm to build local fault diagnosers. These diagnosers are constructed based on global diagnosability analysis of the system, determining the minimal number of residuals required to have the maximum possible global diagnosability in the system. Several criteria are taken into account to select the optimal solutions among the set of solutions. In particular, we used three different metrics: the diagnosability of the diagnosis submodel; the number of residuals; and the amount of analytical redundancy.

Design results on the ADAPT case study demonstrated that the residual search space is considerably smaller for the proposed distributed approach than for the centralized approach. Also, we show, with several design examples, the running of the proposed algorithms and the optimal solutions that they are capable of obtaining for the local diagnosers.

In future work, we will extend the algorithms to find the best distributed diagnoser solution among all the individual local diagnoser solutions, i.e., choose using some local and global criteria which local diagnoser design should be used; this is another search problem. Additionally, in this paper we considered only single faults and a continuous system for the case study, but, in future work, we will study how to extend this solution to multiple fault diagnosis and hybrid systems.

Acknowledgments

A. Bregon's funding for this work was provided by the Spanish MICINN DPI2013-45414-R grant and the program "Ayudas del plan de movilidad de personal investigador - convocatoria 2014" by the University of Valladolid. M. Daigle and I. Roychoudhury's work was funded in part by the NASA System-wide Satefy Assurance Technologies (SSAT) project under the Aviation Safety (AvSafe) Program of the Aeronautics Research Mission Directorate (ARMD).

References

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

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

[3] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1555–1575, Sep. 1995.

[4] A. Bregon, M. Daigle, I. Roychoudhury, G. Biswas, X. Koutsoukos, and B. Pulido, "An event-based distributed diagnosis framework using structural model decomposition," *Artificial Intelligence*, vol. 210, pp. 1–35, May 2014.

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

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

[7] S. Poll, A. Patterson-Hine, J. Camisa, D. Nishikawa, L. Spirkovska, D. Garcia, D. Hall, C. Neukom, A. Sweet, S. Yentus, C. Lee, J. Ossenfort, I. Roychoudhury, M. Daigle, G. Biswas, X. Koutsoukos, and R. Lutz, "Evaluation, selection, and application of model-based diagnosis tools and approaches," in *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, may 2007.

[8] S. Poll, J. de Kleer, R. Abreau, M. Daigle, A. Feldman, D. Garcia, A. Gonzalez-Sanchez, T. Kurtoglu, S. Narasimhan, and A. Sweet, "Third international diagnostics competition – DXC'11," in *Proc. of the 22nd In-*

ternational Workshop on Principles of Diagnosis, Oct. 2011, pp. 267–278.

[9] A. Sweet, A. Feldman, S. Narasimhan, M. Daigle, and S. Poll, "Fourth international diagnostic competition – DXC'13," in *Proc. of the 24th International Workshop on Principles of Diagnosis*, Sep. 2013, pp. 224–229.

[10] Z. Gao and D. W. Ho, "State/noise estimator for descriptor systems with application to sensor fault diagnosis," *Signal Processing, IEEE Transactions on*, vol. 54, no. 4, pp. 1316–1326, 2006.

[11] A. M. Pertew, H. J. Marquez, and Q. Zhao, "Lmi-based sensor fault diagnosis for nonlinear lipschitz systems," *Automatica*, vol. 43, no. 8, pp. 1464–1469, 2007.

[12] S. Wang and F. Xiao, "Ahu sensor fault diagnosis using principal component analysis method," *Energy and Buildings*, vol. 36, no. 2, pp. 147–160, 2004.

[13] M. Zedda and R. Singh, "Gas turbine engine and sensor fault diagnosis using optimization techniques," *Journal of propulsion and power*, vol. 18, no. 5, pp. 1019–1025, 2002.

[14] D. Yu, J. Gomm, and D. Williams, "Sensor fault diagnosis in a chemical process via rbf neural networks," *Control Engineering Practice*, vol. 7, no. 1, pp. 49–55, 1999.

[15] M. Basseville, A. Benveniste, G. Moustakides, and A. Rougee, "Optimal sensor location for detecting changes in dynamical behavior," *IEEE Transactions on Automatic Control*, vol. 32, no. 12, pp. 1067–1075, Dec 1987.

[16] R. Debouk, S. Lafortune, and D. Teneketzis, "On an optimization problem in sensor selection," *Discrete Event Dynamic Systems*, vol. 12, no. 4, pp. 417–445, 2002.

[17] I. Roychoudhury, G. Biswas, and X. Koutsoukos, "Designing distributed diagnosers for complex continuous systems," *IEEE Transactions on Automation Science and Engineering*, vol. 6, no. 2, pp. 277–290, Apr. 2009.

[18] M. Daigle, A. Bregon, G. Biswas, X. Koutsoukos, and B. Pulido, "Improving multiple fault diagnosability using possible conflicts," in *Proceedings of the 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Aug. 2012, pp. 144–149.

[19] P. J. Mosterman and G. Biswas, "Diagnosis of continuous valued systems in transient operating regions," *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 29, no. 6, pp. 554–565, 1999.

[20] M. J. Daigle, X. Koutsoukos, and G. Biswas, "A qualitative event-based approach to continuous systems diagnosis," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 4, pp. 780–793, Jul. 2009.

[21] M. Daigle, X. Koutsoukos, and G. Biswas, "Distributed diagnosis in formations of mobile robots," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 353–369, Apr. 2007.

[22] M. Daigle, "A qualitative event-based approach to fault diagnosis of hybrid systems," Ph.D. dissertation, Vanderbilt University, 2008.

[23] D. C. Karnopp, D. L. Margolis, and R. C. Rosenberg, *Systems Dynamics: Modeling and Simulation of Mechatronic Systems*. New York: John Wiley & Sons, Inc., 2000.

[24] M. Daigle, I. Roychoudhury, and A. Bregon, "Diagnosability-based sensor placement through structural model decomposition," in *Proceedings of the Second European Conference of the Prognostics and Health Management Society 2014*, Jul. 2014, pp. 33–46.

[25] T. Kurtoglu, S. Narasimhan, S. Poll, D. Garcia, L. Kuhn, J. de Kleer, A. van Gemund, and A. Feldman, "First international diagnosis competition – DXC'09," in *Proceedings of 20th International Workshop on Principles of Diagnosis*, Jun. 2009, pp. 383–396.

[26] M. Daigle, A. Bregon, and I. Roychoudhury, "Qualitative Event-based Diagnosis with Possible Conflicts: Case Study on the Third International Diagnostic Competition," in *Proceedings of the 22nd International Workshop on Principles of Diagnosis*, Murnau, Germany, Oct. 2011, pp. 285–292.

[27] M. Daigle, I. Roychoudhury, and A. Bregon, "Qualitative event-based diagnosis with possible conflicts: Case study on the fourth international diagnostic competition," in *Proceeedings of the 24th International Workshop on Principles of Diagnosis*, Oct. 2013, pp. 230–235.

BIOGRAPHY

Anibal Bregon *received his B.Sc., M.Sc., and Ph.D. degrees in Computer Science from the University of Valladolid, Spain, in 2005, 2007, and 2010, respectively. From September 2005 to June 2010, he was Graduate Research Assistant with the Intelligent Systems Group at the University of Valladolid, Spain. He has been visiting researcher at the Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA; the Dept. of Electrical Engineering, Linkoping University, Linkoping, Sweden; and the Diagnostics and Prognostics Group, NASA Ames Research Center, Mountain View, CA, USA. Since September 2010, he has been Assistant Professor and Research Scientist at the Department of Computer Science from the University of Valladolid. Dr. Bregon is a member of the Prognostics and Health Management Society and the IEEE. His current research interests include model-based reasoning for diagnosis, prognostics, health-management, and distributed diagnosis of complex physical systems.*

Matthew Daigle *received the B.S. degree in Computer Science and Computer and Systems Engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, TN, in 2006 and 2008, respectively. From September 2004 to May 2008, he was a Graduate Research Assistant with the Institute for Software Integrated Systems and Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN. From June 2008 to December 2011, he was an Associate Scientist with the University of California, Santa Cruz, at NASA Ames Research Center. Since January 2012, he has been with NASA Ames Research Center as a Research Computer Scientist. His current research interests include physics-based modeling, model-based diagnosis and prognosis, sim-*

ulation, and hybrid systems. Dr. Daigle is a member of the Prognostics and Health Management Society and the IEEE.

Indranil Roychoudhury *received the B.E. (Hons.) degree in Electrical and Electronics Engineering from Birla Institute of Technology and Science, Pilani, Rajasthan, India in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, Tennessee, USA, in 2006 and 2009, respectively. Since August 2009, he has been with SGT, Inc., at NASA Ames Research Center as a Computer Scientist. Dr. Roychoudhury is a member of the Prognostics and Health Management Society and the IEEE. His research interests include hybrid systems modeling, model-based diagnostics and prognostics, distributed diagnostics and prognostics, and Bayesian diagnostics of complex physical systems.*