Diagnosability-Based Sensor Placement through Structural Model Decomposition

Matthew Daigle[1], Indranil Roychoudhury[2], and Anibal Bregon[3]

[1] *NASA Ames Research Center, Moffett Field, California, 94035, USA*
matthew.j.daigle@nasa.gov

[2] *SGT Inc., NASA Ames Research Center, Moffett Field, California, 94035, USA*
indranil.roychoudhury@nasa.gov

[3] *Department of Computer Science, University of Valladolid, Valladolid, Spain*
anibal@infor.uva.es

ABSTRACT

Systems health management, and in particular fault diagnosis, is important for ensuring safe, correct, and efficient operation of complex engineering systems. The performance of an online health monitoring system depends critically on the available sensors of the system. However, the set of selected sensors is subject to many constraints, such as cost and weight, and hence, these sensors must be selected judiciously. This paper presents an offline design-time sensor placement approach for complex systems. Our diagnosis method is built upon the analysis of model-based residuals, which are computed using structural model decomposition. Sensor placement in this framework manifests as a residual selection problem, and we aim to find the set of residuals that achieves single-fault diagnosability of the system, uses the minimum number of sensors, and corresponds to the best model decomposition for the best distribution of the diagnosis system. We present a set of algorithms for solving this problem and compare their performance in terms of computational complexity and optimality of solutions. We demonstrate the approach using a benchmark multi-tank system.

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). The performance of the fault diagnosis system depends on the available sensors from which diagnostic information can be extracted. However, the set of selected sensors are subject to many constraints, such as cost and weight, and hence, these sensors must be selected judiciously.

For fault isolation, a valid placement of sensors should be one in which the system is diagnosable, i.e., all single faults can be uniquely isolated from each other, which is a design metric for diagnostic performance (Narasimhan, Mosterman, & Biswas, 1998). We utilize a fault isolation framework that is based on the analysis of model-based residuals, where each residual is computed as the difference between a measured sensor output and the predicted value of that sensor output. Local models of the system, that are used to make predictions of measured outputs, are generated by decomposing the global model of the system using structural model decomposition (Roychoudhury, Daigle, Bregon, & Pulido, 2013). Therefore, the problem of sensor placement is directly related to one of residual selection.

In this paper, we formulate the sensor placement problem and establish its search space through the novel concept of a complete residual set, based on structural model decomposition. We present three algorithms for solving this sensor placement problem and compare their performance in terms of computational complexity and optimality of solutions. The different algorithms are: (*i*) exhaustive search, (*ii*) stochastic search, and (*iii*) structured search. The exhaustive search is a brute force search over the residual space, and so guarantees optimality but is not scalable. The stochastic search selects random residual sets and modifies them randomly to try to improve the current set of candidate solutions. The structured search algorithm uses knowledge of what solutions are preferred in order to search a reduced space in a structured fashion. We demonstrate the approach using a benchmark multi-

tank system (Daigle, Bregon, Biswas, Koutsoukos, & Pulido, 2012). In this work, we focus on continuous systems and adopt the single fault assumption.

This paper is organized as follows. Section 2 presents related work to set the context for our contributions. Section 3 provides the necessary background information on structural model decomposition and our qualitative fault isolation framework. The problem formulation, which defines the problem and establishes its search space, is presented in Section 4. The diagnosability-based measurement selection approach and the three algorithms are described in Section 5. Experimental results are provided in Section 6. Section 7 concludes the paper and discusses future work.

2. RELATED WORK

Efficient solutions to the sensor placement problem have been explored before. Our work is in contrast to other approaches present in literature (Basseville, Benveniste, Moustakides, & Rougee, 1987; Debouk, Lafortune, & Teneketzis, 2002; Roychoudhury, Biswas, & Koutsoukos, 2009) 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 (Basseville et al., 1987), the authors propose an approach for optimal sensor location to increase the fault detection performance in dynamic systems using statistical tests. In (Debouk et al., 2002) 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. Our decision in favor of distributed approaches is influenced by the fact that the design of fault diagnosers can have consequences in terms of computational efficiency, scalability, single points of failure, and the quickness of fault diagnosis, among others (Roychoudhury et al., 2009). For example, centralized diagnosis approaches suffer from single points of failure, large computational complexity, and scalability issues. Decomposing the diagnosis problem can address some or all of these issues.

Unlike the related approach of (Roychoudhury et al., 2009), another focus of our work is in the use of structural information to determine the best sensor locations. Several previous papers make use of structural information for solving the sensor placement problem (Krysander & Frisk, 2008; Rosich, 2012; Travé-Massuyès et al., 2006; Said & Djamel, 2013). The use of structural information allows to efficiently solve this problem for large and nonlinear differential-algebraic models. In (Krysander & Frisk, 2008) the authors propose new a method, using Dulmage-Mendelson decomposition, for computing which sensors to add to obtain maximum fault detectability and isolability. A related approach is proposed in (Travé-Massuyès et al., 2006), but following a different strategy. Instead of computing which sensors to add to obtain a certain isolability performance, in (Travé-Massuyès et al., 2006) the problem is solved by hypothesizing sensors, then computing Analytical Redundancy Relations (ARR) with all possible causalities, and then obtaining the isolability properties. In this sense, our approach is more similar to the approach of (Krysander & Frisk, 2008), since we add sensors looking for maximum diagnosability and then decompose the system to look for the smaller submodels to obtain that maximum diagnosability. However, our approach is different to both, since we include qualitative and temporal information within our models, which improves diagnosability; and second, the approach in (Travé-Massuyès et al., 2006) only allows solutions where residuals are computed by using minimal submodels.

Other approaches in the literature consider causal information within the system model (Raghuraj, Bhushan, & Rengaswamy, 1999; Rosich, Frisk, Aslund, Sarrate, & Nejjari, 2012). In (Raghuraj et al., 1999), the authors use a directed graph and algorithms based on the graph to look for the optimal sensor location to ensure observability and fault resolution. Also, the authors discuss the possibility of including signs in the graph. However, unlike the approach presented in this paper, signs are not included in the algorithms. Another difference against our approach is that they only consider residuals computed using the global system model. In (Rosich et al., 2012), the authors only allow residuals computed from minimal submodels and temporal information is not included.

One of the main problems of the structural approach to sensor placement, especially when a large number of feasible sensor locations is available, is that the computational effort to look for the optimal solution could be huge. In (Eriksson, Krysander, & Frisk, 2012) the authors use a quantitative diagnosability analysis to optimize sensor placement for fault diagnosis. In (Casillas, Puig, Garza-Castañon, & Rosich, 2013) genetic algorithms are used for the same task. Unlike the previous approaches, in (Frisk, Krysander, & Åslund, 2009) the authors use analytical equations as a solution which can handle models where structural approaches fail, however, it is limited only to linear differential-algebraic systems, which restricts severely its applicability to practical systems.

3. BACKGROUND

In this section, we first describe our approach to structural model decomposition, which, given a global system model, creates local models of system behavior. We then describe our model-based diagnostic framework that is based on analysis of residuals computed using these local models.

3.1. System Modeling

We adopt here the structural model decomposition framework described in (Roychoudhury et al., 2013). In the following,

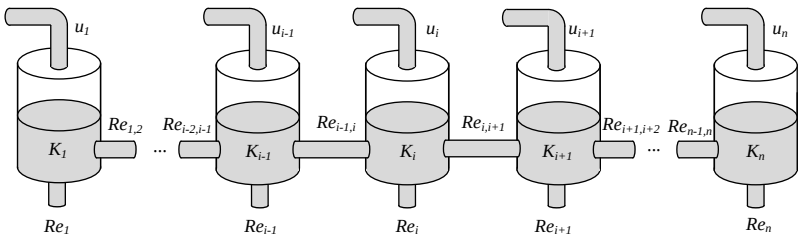


Figure 1. Tank system schematic.

we review the main details and and refer the interested reader to (Roychoudhury et al., 2013) for additional explanation. 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 reduce the structural complexity of the equations.

Throughout the paper, we use a benchmark multi-tank system as a running example. The system consists of n tanks connected serially, as shown in Fig. 1. For each tank i, where $i \in [1, n]$, u_i denotes the input flow, m_i denotes the liquid mass, p_i denotes the tank pressure, q_i denotes the mass flow out of the drain pipe, K_i denotes the tank capacitance, and Re_i denotes the drain pipe resistance. For adjacent tanks i and $i+1$, $q_{i,i+1}$ denotes the mass flow from tank i to tank $i+1$ through the connecting pipe, and $Re_{i,i+1}$ is the connecting pipe resistance. The constraints for tank i are as follows:

$$\begin{aligned}
\dot{m}_i &= u_i + q_{i-1,i} - q_i - q_{i,i+1},\\
m_i &= \int_{t_0}^{t} \dot{m}_i dt,\\
p_i &= \frac{1}{K_i} m_i,\\
q_i &= \frac{1}{Re_i} p_i,\\
q_{i,i+1} &= \frac{1}{Re_{i,i+1}} (p_i - p_{i+1}).
\end{aligned}$$

For tank 1, $q_{0,1} = 0$, and for tank n, $q_{n,n+1} = 0$.

The measurements corresponding to p_i, q_i, and $q_{i,i+1}$ are p_i^*, q_i^*, and $q_{i,i+1}^*$ and are described by the following constraints:

$$\begin{aligned}
p_i^* &= p_i,\\
q_i^* &= q_i,\\
q_{i,i+1}^* &= q_{i,i+1}.
\end{aligned}$$

Example 1. For a three-tank system measuring the output flows, the model $\mathcal{M}^*$ is represented by the variable sets $X = \{m_1, m_2, m_3\}$, $\Theta = \{K_1, K_2, K_3, Re_1, Re_2, Re_3, Re_{1,2}, Re_{2,3}\}$, $U = \{u_1, u_2, u_3\}$, $Y = \{p_1^*, p_2^*, p_3^*, q_1^*, q_2^*, q_3^*, q_{1,2}^*, q_{2,3}^*\}$, and $A = \{\dot{m}_1, \dot{m}_2, \dot{m}_2, p_1, p_2, p_3, q_1, q_2, q_3\}$; and the set of constraints $C = \{c_1, c_2, \ldots, c_{22}\}$, where the constraints are given as follows:

$$\dot{m}_1 = u_1 - q_1 - q_{1,2}, \quad (c_1)$$
$$\dot{m}_2 = u_2 + q_{1,2} - q_2 - q_{2,3}, \quad (c_2)$$
$$\dot{m}_3 = u_3 + q_{2,3} - q_3, \quad (c_3)$$
$$m_1 = \int_{t_0}^{t} \dot{m}_1 dt, \quad (c_4)$$
$$m_2 = \int_{t_0}^{t} \dot{m}_2 dt, \quad (c_5)$$
$$m_3 = \int_{t_0}^{t} \dot{m}_3 dt, \quad (c_6)$$
$$p_1 = \frac{1}{K_1} m_1, \quad (c_7)$$
$$p_2 = \frac{1}{K_2} m_2, \quad (c_8)$$
$$p_3 = \frac{1}{K_3} m_3, \quad (c_9)$$
$$q_1 = \frac{1}{Re_1} p_1, \quad (c_{10})$$
$$q_2 = \frac{1}{Re_2} p_2, \quad (c_{11})$$
$$q_3 = \frac{1}{Re_3} p_3, \quad (c_{12})$$
$$q_{1,2} = \frac{1}{Re_{1,2}} (p_1 - p_2), \quad (c_{13})$$
$$q_{2,3} = \frac{1}{Re_{2,3}} (p_2 - p_3), \quad (c_{14})$$
$$p_1^* = p_1, \quad (c_{15})$$

$$p_2^* = p_2, \quad (c_{16})$$
$$p_3^* = p_3, \quad (c_{17})$$
$$q_1^* = q_1, \quad (c_{18})$$
$$q_2^* = q_2, \quad (c_{19})$$
$$q_3^* = q_3, \quad (c_{20})$$
$$q_{1,2}^* = q_{1,2}, \quad (c_{21})$$
$$q_{2,3}^* = q_{2,3}. \quad (c_{22})$$

Here, the * superscript is used to denote a measured value of a physical variable, e.g., p_1 is pressure and p_1^* is the measured pressure. Since p_1 is used to compute other variables, it cannot belong to Y and a separation of the variables is required.

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 states that (*i*) input or parameter variables cannot be the dependent variables in the causal assignment, (*ii*) a measured variable can be used as the dependent variable, and (*iii*) every variable, which 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.

For the n-tank system, the causal constraints for tank i are as follows:

$$\dot{m}_i := u_i + q_{i-1,i} - q_i - q_{i,i+1},$$
$$m_i := \int_{t_0}^{t} \dot{m}_i dt,$$
$$p_i := \frac{1}{K_i} m_i,$$
$$q_i := \frac{1}{Re_i} p_i,$$
$$q_{i,i+1} := \frac{1}{Re_{i,i+1}} (p_i - p_{i+1}),$$
$$p_i^* := p_i,$$
$$q_i^* := q_i,$$
$$q_{i,i+1}^* := q_{i,i+1}.$$

Example 2. The causal model $\mathcal{M}$ is represented by the same variables and constraints as $\mathcal{M}^*$, along with the set of causal assignments $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_{22}\}$, as given below:

$$\dot{m}_1 := u_1 - q_1 - q_{1,2}, \quad (\alpha_1)$$
$$\dot{m}_2 := u_2 + q_{1,2} - q_2 - q_{2,3}, \quad (\alpha_2)$$
$$\dot{m}_3 := u_3 + q_{2,3} - q_3, \quad (\alpha_3)$$
$$m_1 := \int_{t_0}^{t} \dot{m}_1 dt, \quad (\alpha_4)$$
$$m_2 := \int_{t_0}^{t} \dot{m}_2 dt, \quad (\alpha_5)$$
$$m_3 := \int_{t_0}^{t} \dot{m}_3 dt, \quad (\alpha_6)$$
$$p_1 := \frac{1}{K_1} m_1, \quad (\alpha_7)$$
$$p_2 := \frac{1}{K_2} m_2, \quad (\alpha_8)$$
$$p_3 := \frac{1}{K_3} m_3, \quad (\alpha_9)$$
$$q_1 := \frac{1}{Re_1} p_1, \quad (\alpha_{10})$$
$$q_2 := \frac{1}{Re_2} p_2, \quad (\alpha_{11})$$
$$q_3 := \frac{1}{Re_3} p_3, \quad (\alpha_{12})$$
$$q_{1,2} := \frac{1}{Re_{1,2}} (p_1 - p_2), \quad (\alpha_{13})$$
$$q_{2,3} := \frac{1}{Re_{2,3}} (p_2 - p_3), \quad (\alpha_{14})$$
$$p_1^* := p_1, \quad (\alpha_{15})$$
$$p_2^* := p_2, \quad (\alpha_{16})$$
$$p_3^* := p_3, \quad (\alpha_{17})$$
$$q_1^* := q_1, \quad (\alpha_{18})$$
$$q_2^* := q_2, \quad (\alpha_{19})$$
$$q_3^* := q_3, \quad (\alpha_{20})$$
$$q_{1,2}^* := q_{1,2}, \quad (\alpha_{21})$$
$$q_{2,3}^* := q_{2,3}. \quad (\alpha_{22})$$

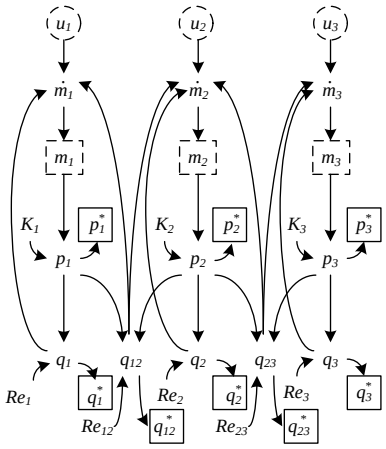


Figure 2. Causal graph of the three-tank system.

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.

3.2. 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 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 in which each residual responds to only a subset of the faults, increasing diagnosability (Daigle et al., 2012).

Under this approach, given a (global) model, we can create (local) submodels that use as additional inputs values from the sensors (Roychoudhury et al., 2013). 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 causal submodel from a causal model is given as Algorithm 1 (Roychoudhury et al., 2013). Given a causal model $\mathcal{M}$, and an output variable to be computed y, the `GenerateSubmodel` algorithm derives a causal submodel $\mathcal{M}_i$ that computes y using as local inputs only variables from $U^* = U \cup (Y - \{y\})$. 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 Subroutine 2 (`GetBestConstraint` subroutine) to find the constraint that should be used to resolve v in the minimal way.

The `GetBestConstraint` subroutine (which has been updated from (Roychoudhury et al., 2013)) 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^*); 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^*; or such a constraint may be one that computes some y^* through some v' in an algebraic relation. 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. A preferences list, P, is used to break ties if multiple minimal constraints exist to resolve v.

We assume that the differential constraints in the model are always in integral causality. We assume also that the model $\mathcal{M}$ to be decomposed is free from algebraic loops (which will prevent Algorithm 1 from terminating), otherwise, the constraints may be arbitrarily complex and nonlinear. However, nonlinear constraints may not be possible in all causalities. If causality must be changed in order for the decomposition to proceed, there must be an expression for the constraint in the new causal form. If some constraints are not available in all possible causalities, then this may restrict the possible model decompositions.

Algorithm 1 $\mathcal{M}_i = \texttt{GenerateSubmodel}(\mathcal{M}, U^*, V^*)$

$V_i \leftarrow V^*$
$C_i \leftarrow \varnothing$
$\mathcal{A}_i \leftarrow \varnothing$
$variables \leftarrow V^*$
while $variables \neq \varnothing$ **do**
  $v \leftarrow \texttt{pop}(variables)$
  $c \leftarrow \texttt{GetBestConstraint}(v, V_i, U^*, \mathcal{A})$
  $C_i \leftarrow C_i \cup \{c\}$
  $\mathcal{A}_i \leftarrow \mathcal{A}_i \cup \{(c, v)\}$
  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 $c = \texttt{GetBestConstraint}(v, V_i, U^*, \mathcal{A})$

$C \leftarrow \varnothing$
$c_v \leftarrow$ **find** c **where** $(c, v) \in \mathcal{A}$
if $V_{c_v} \subseteq V_i \cup U^*$ **then**
  $C \leftarrow C \cup \{c_v\}$
end if
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** $V_{c_y} \subseteq V_i \cup U^* \cup \Theta$ **then**
    $C \leftarrow C \cup \{c_y\}$
  end if
end for
for all $y \in Y \cap U^*$ **do**
  $c_y \leftarrow$ **find** c **where** $(c, y) \in \mathcal{A}$
  $V' \leftarrow V_{c_y} - \{y\}$
  for all $v' \in V'$ **do**
    $c_{v'} \leftarrow$ **find** c **where** $(c, v') \in \mathcal{A}$
    if $v \in V_{c_{v'}}$ **and** $V_{c_y} \subseteq \{v\} \cup U^* \cup \Theta$ **then**
      $C \leftarrow C \cup \{c_{v'}\}$
    end if
  end for
end for
if $C = \varnothing$ **then**
  $c \leftarrow c_v$
else if $c_v \in C$ **then**
  $c \leftarrow c_v$
else
  $C' \leftarrow C$
  for all $c_1, c_2 \in C$ **where** $c_1 \neq c_2$ **do**
    $y_1 \leftarrow$ **find** y **where** $(c_1, y_1) \in \mathcal{A}$
    $y_2 \leftarrow$ **find** y **where** $(c_2, y_2) \in \mathcal{A}$
    if $(y_1 \triangleleft y_2) \in P$ **then**
      $C' \leftarrow C' - \{c_1\}$
    end if
  end for
  $c \leftarrow \texttt{first}(C')$
end if

3.3. Qualitative Fault Isolation

As mentioned in Section 1, the goal of this work is to solve the sensor placement problem such that all single faults can be uniquely isolated from each other. The solution of this problem depends on the diagnosis framework chosen. In this section, we briefly present our fault isolation approach. For details, please refer to (Mosterman & Biswas, 1999; Bregon et al., 2014).

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 three-tank system in Fig. 1, the complete fault set F consists of $\{K_1^-, K_1^+, K_2^-, K_2^+, K_3^-, K_3^+, Re_1^-, Re_1^+, Re_2^-, Re_2^+, Re_3^-, Re_3^+, Re_{1,2}^-, Re_{1,2}^+, Re_{2,3}^-, Re_{2,3}^+\}$.

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 the previous subsection, we see that there are several potential submodels that can compute $\hat{y}$, depending on what local inputs are selected. 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* (Mosterman & Biswas, 1999).

Definition 7 (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* (Daigle, Koutsoukos, & Biswas, 2007).

Definition 8 (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 (Mosterman & Biswas, 1999) (TCG) is obtained, and

the algorithms described in (Daigle, 2008) may be used to automatically derive the signatures and orderings.[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 9 (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 $\Omega_{f,R}$.

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

Definition 10 (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 11 (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[2] (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 12 (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.

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

Definition 13 (Diagnosability)**.** A diagnosis model $\mathcal{S} = (F, Y, R, L_{F,R})$ is *diagnosable* if and only if $(\forall f_i, f_j \in F) f_i \neq f_j \implies f_i \nsim_R f_j$.

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.

[1] TCGs can also be derived directly from bond graphs (Karnopp, Margolis, & Rosenberg, 2000). Our modeling approach is more general in that it is not restricted to system topologies imposed by bond graphs.

[2] 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. PROBLEM FORMULATION

The problem we are trying to solve is one of sensor placement for diagnosability. In our diagnostic framework, diagnosability is based on residuals, and so the sensor placement problem manifests as a residual selection problem. For each set of sensors, there are many potential residuals that can be selected to achieve diagnosability. A solution to the problem is a selection of residuals that achieves diagnosability; an optimal solution is one that satisfies some given criteria the best.

As described in Section 3, residuals are defined from submodel outputs. Given a model $\mathcal{M}$, there are many submodels that can be defined, and residuals can be derived from each of these. Clearly, this residual space is exceedingly large. However, many of these residuals are not actually unique, i.e., there may be two submodels that use the exact same computations to produce two different residuals; in this case, the residuals are equivalent. We express this property through the concept of *residual equivalence*.

Definition 14 (Residual Equivalence)**.** Given causal submodels $\mathcal{M}_i$ with inputs U_i and output $y_i \in Y_i$, and $\mathcal{M}_j$ with inputs U_j and output $y_j \in Y_j$, the residuals r_i computed from y_i and r_j computed from y_j are equivalent, denoted as $r_i \equiv r_j$ if the causal constraints used to compute y_i are the same as the causal constraints to compute y_j.

We need not consider solutions that contain residuals that are equivalent, and this reduces the search space. We refer to such a residual set as *minimal*.

Definition 15 (Minimal Residual Set)**.** A residual set R is *minimal* if there are no two residuals $r_i \in R$ and $r_j \in R$, $i \neq j$, where $r_i \equiv r_j$.

In fact we need only to search over the space of unique residuals. For a given sensor set, we can define the corresponding *complete residual set*, i.e., the largest set of residuals for a sensor set that is minimal.

Definition 16 (Complete Residual Set)**.** For a set of sensor outputs Y, the *complete residual set* is the minimal residual set R_Y such that there is no residual for an output in $y \in Y$, r_y, such that $R \cup \{r_y\}$ is also minimal.

The complete residual set contains, for a given set of sensors, every possible way of computing residuals for those sensors.

So, the space of the residual selection problem is defined by all combinations of residuals in the unique residual set. We can find the complete residual set by using the model decomposition algorithm. As described in Section 3, to compute a submodel we must define the available local input set U^* and the local output set V^*. For an ouput $y \in Y$, U^* must consist of U and elements from $Y - \{y\}$. For example, say we have a three-tank system where $Y = \{q_1^*, q_2^*, q_3^*\}$. Table 1 lists all possible U^* to compute each output y. Fig. 3 show the causal

$\mathcal{M}_i$	U_i^*	V_i^*	U_i
$\mathcal{M}_1$	$\{u_1, u_2, u_3\}$	$\{q_1^*\}$	$\{u_1, u_2, u_3\}$
$\mathcal{M}_2$	$\{u_1, u_2, u_3, q_2^*\}$	$\{q_1^*\}$	$\{u_1, q_2^*\}$
$\mathcal{M}_3$	$\{u_1, u_2, u_3, q_3^*\}$	$\{q_1^*\}$	$\{u_1, u_2, q_3^*\}$
$\mathcal{M}_4$	$\{u_1, u_2, u_3, q_2^*, q_3^*\}$	$\{q_1^*\}$	$\{u_1, q_2^*\}$
$\mathcal{M}_5$	$\{u_1, u_2, u_3\}$	$\{q_2^*\}$	$\{u_1, u_2, u_3\}$
$\mathcal{M}_6$	$\{u_1, u_2, u_3, q_1^*\}$	$\{q_2^*\}$	$\{u_2, u_3, q_1^*\}$
$\mathcal{M}_7$	$\{u_1, u_2, u_3, q_3^*\}$	$\{q_2^*\}$	$\{u_1, u_2, q_3^*\}$
$\mathcal{M}_8$	$\{u_1, u_2, u_3, q_1^*, q_3^*\}$	$\{q_2^*\}$	$\{u_2, q_1^*, q_3^*\}$
$\mathcal{M}_9$	$\{u_1, u_2, u_3\}$	$\{q_3^*\}$	$\{u_1, u_2, u_3\}$
$\mathcal{M}_{10}$	$\{u_1, u_2, u_3, q_1^*\}$	$\{q_3^*\}$	$\{u_2, u_3, q_1^*\}$
$\mathcal{M}_{11}$	$\{u_1, u_2, u_3, q_2^*\}$	$\{q_3^*\}$	$\{u_3, q_2^*\}$
$\mathcal{M}_{12}$	$\{u_1, u_2, u_3, q_1^*, q_2^*\}$	$\{q_3^*\}$	$\{u_3, q_2^*\}$

Table 1. Single-output Submodels

graphs for submodels $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, and $\mathcal{M}_4$. The thicker, green arrows in Fig. 3 indicate causal assignments that were reversed to accommodate local inputs.

There are $2^{|Y|-1}$ possible subsets of $Y - \{y\}$ from which to define U^*, and hence $|Y|2^{|Y|-1}$ residuals. However, this residual set may not be minimal. The model decomposition algorithm finds the minimal submodel to compute the given V^* using U^*, therefore, for a given V^* and two different U^*, the derived submodel may have the same U_i and the same causal constraints. This occurs for the example in Table 1. Here, the q_1^* residuals from $\mathcal{M}_2$ and $\mathcal{M}_4$ are equivalent. Since the U_i are the same, the submodel must be using the same constraints to compute q_1^* (see Fig. 3). Similarly, the q_3^* residuals from $\mathcal{M}_{11}$ and $\mathcal{M}_{12}$ are equivalent. So, in this case, the complete residual set size is less than $|Y|2^{|Y|-1}$.

So, a solution to the problem will be a selection of residuals from the complete residual set that achieves diagnosability. Among these solutions, we desire only those that require the minimum number of sensors (where measured values may be used to compute residuals and/or as local inputs to submodels). We may prefer some solutions over others for a variety of reasons. We define a relational operator $\succ$ for solutions, describing which solutions are preferred over others and thus obtaining a notion of optimality for solutions. The $\succ$ 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. For a model $\mathcal{M}$, fault set F, and sensor set Y, the problem is to find a set of residuals R_i such that there is no other $R_j \neq R_i$ where $R_j \succ R_i$.

5. APPROACH

In this section, we introduce three different algorithms to solve the problem stated in Section 4. For validation purposes, we describe an exhaustive search algorithm. We describe also a stochastic search algorithm and a structured search algorithm.

Before defining the $\succ$ operator it is first important to note that residuals can be associated with submodels larger than those computing only themselves. Consider Table 1. For

$\mathcal{M}_i$	U_i	Y_i
$\mathcal{M}_1$	$\{u_1, u_2, u_3\}$	$\{q_1^*, q_2^*, q_3^*\}$
$\mathcal{M}_2$	$\{u_1, u_2, q_3^*\}$	$\{q_1^*, q_2^*\}$
$\mathcal{M}_3$	$\{u_2, u_3, q_1^*\}$	$\{q_2^*, q_3^*\}$
$\mathcal{M}_4$	$\{u_1, q_2^*\}$	$\{q_1^*\}$
$\mathcal{M}_5$	$\{u_2, q_1^*, q_3^*\}$	$\{q_2^*\}$
$\mathcal{M}_6$	$\{u_3, q_2^*\}$	$\{q_3^*\}$

Table 2. Multi-Output Submodels

any U_i that can compute more than one residual, the submodels computing these residuals can be easily merged into a multi-output submodel computing all the residuals. This submodel can be computed by taking the union of the variable and constraint sets of the individual submodels. For example, in Table 1, we see that $\mathcal{M}_1$, $\mathcal{M}_5$, and $\mathcal{M}_9$ all have the input set $\{u_1, u_2, u_3\}$; merging these submodels recovers the global model. Also, the input set $\{u_1, u_2, q_3^*\}$ can be used to compute both a residual for q_1^* and one for q_2^* ($\mathcal{M}_3$ and $\mathcal{M}_7$, respectively). So for a given residual set, there is also an associated submodel set, defined by the sets of inputs used to compute the residuals in the set. All the associated submodels for the example in Table 1 are given in Table 2.

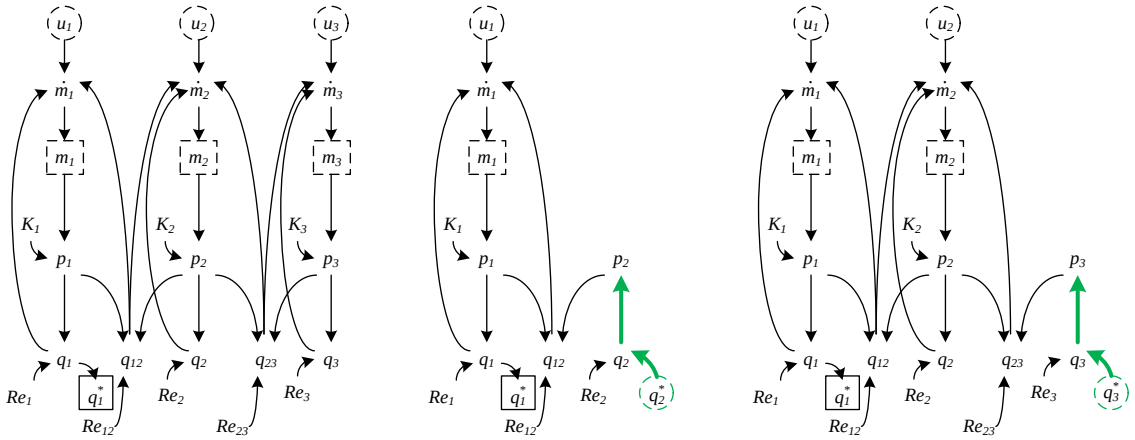
For a given set of sensors such that diagnosability can be achieved, what we desire is a solution that corresponds to some notion of the best model decomposition (for the submodel set associated with the residual set). The more independent submodels that are used, the more distributed the solution becomes. Since the submodels are computationally independent, they can be executed in parallel and thus naturally take advantage of distributed computational paradigms. Further, model decomposition can lead to improved diagnosability (Daigle et al., 2012).

We define the $\succ$ operator using five metrics: (*i*) diagnosability, (*ii*) the number of sensors used, (*iii*) the minimality of the involved submodel set, (*iv*) the number of residuals per submodel, and (*v*) the total number of residuals. We explain each of these in turn, starting with the minimality of a submodel set, which is defined as follows.

Definition 17. For a given set of residuals R, the corresponding submodel set M is minimal if for any $\mathcal{M}_i \in M$, there is no other $\mathcal{M}_j \in M$, $\mathcal{M}_i \neq \mathcal{M}_j$, that can be created by decomposing that submodel.

We do not prefer such solutions because they are likely to include residuals that do not improve diagnosability. For example, if the global model is in the submodel set, it is unlikely that adding additional submodels, for which there are already residuals for their outputs, will add diagnosability, since there is much redundant information in a residual for the same output over two different submodels.

We prefer fewer residuals per submodel because this implies a greater level of decomposition, and we prefer fewer residuals, since that implies the solutions are minimal, i.e., they do not include additional residuals that are not needed to obtain



(a) $\mathcal{M}_1$ Causal Graph (b) $\mathcal{M}_2$, $\mathcal{M}_4$ Causal Graph (c) $\mathcal{M}_3$ Causal Graph

Figure 3. Causal graphs for submodels $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, and $\mathcal{M}_4$.

Algorithm 3 $R^* = \texttt{ExhaustiveSearch}(R)$

$\mathcal{R} \leftarrow \texttt{combos}(R)$
$R^* \leftarrow \mathcal{R}_1$
for all $R_i \in \mathcal{R}$ **do**
  if $R_i \succ R^*$ **then**
    $R^* \leftarrow R_i$
  end if
end for

diagnosability.

We next present the three algorithms, in which the inputs will be the complete residual sets as defined here. Other residual sets, computed using different methods, may also be used with no or little changes to the algorithms.

5.1. Exhaustive Search

As described in Section 4, the search space is defined by the complete residual set. The exhaustive search algorithm is shown as Algorithm 3. The `combos` function returns all possible combinations of the residual set (this is the same as the power set of R, excluding the empty set). The algorithm tries each candidate solution, keeping track of the best solution observed so far. Because it tries all possibilities, it is guaranteed to find the optimal solution, so it can be used to validate the solutions of the other algorithms. However, it is not scalable, as it must consider in the worst case $|Y|2^{|Y|-1}$ candidate solutions.

5.2. Stochastic Search

The stochastic search sacrifices optimality for scalability. It is given as Algorithm 4. It begins with k random candidate solutions generated using the `randomCombos` function. Beginning with multiple solutions rather than a single solution helps reduce the chances of getting stuck in a local minimum. For each candidate solution, it randomly adds or deletes a

Algorithm 4 $R^* = \texttt{StochasticSearch}(R, k, N)$

$\mathcal{R} \leftarrow \texttt{randomCombos}(R, k)$
for $i = 1$ **to** N **do**
  for all $R_i \in \mathcal{R}$ **do**
    $R_i' \leftarrow \texttt{randomModify}(R_i)$
    if $R_i' \succ R_i$ **then**
      $R_i \leftarrow R_i'$
    end if
  end for
end for
$R^* \leftarrow \mathcal{R}_1$
for all $R_i \in \mathcal{R}$ **do**
  if $R_i \succ R^*$ **then**
    $R^* \leftarrow R_i$
  end if
end for

residual using the `randomModify` function, and, if this improves the solution, then this solution is kept. This process repeats for N iterations, therefore it explores only kN solutions, where both k and N are selected by the user. For larger search spaces, it is more likely to find a good solution with larger values of k and N. If there are many good solutions in the solution space, then this algorithm is likely to find at least one of them, given enough iterations.

5.3. Structured Search

The exhaustive and stochastic search algorithms represent approaches at two opposite ends of the spectrum. We want scalability as well as guarantees of optimality. We can do this by searching through the residual space in a structured way, trying to avoid parts of the search space that we know will not contain optimal solutions.

First, we note that we desire solutions with the minimum number of required sensors. Therefore, as a first stage in the algorithm, we try to find minimum sensor sets that can provide complete diagnosability. To do this, we start with single sensor solutions, one for each potential sensor. The only

available residuals for single-sensor solutions are those from the global model. If any of these are diagnosable, then we have found an optimal solution. Otherwise, for each of these candidates, we add a second sensor, and check if the candidate solution containing all available residuals for that sensor set provide complete diagnosability. If so, we add this solution to a set of solutions to analyze later. We continue in this manner, adding sensors, until diagnosability is achieved, thus resulting in initial minimum sensor solutions.

As a second stage, for each of these initial minimum sensor solutions, we select residuals, in a structured way, for the given sensor set. We select residual sets using knowledge of what kind of solutions we consider to be optimal. First, instead of selecting residuals, we select submodels, and for the selected submodels, select all associated residuals for the given sensor set. Since the submodel space is much smaller than the residual space, this shrinks the search space significantly. Second, we know that for a given sensor set, diagnosability cannot be achieved if any one sensor is removed, therefore, we must consider only solutions in which residuals for each sensor are provided. Therefore, we try only combinations that cover all the sensors. So, for each sensor, we select a submodel that computes a residual for that sensor. Over those combinations there are much fewer to consider.

For example, consider again Table 1. Assume the sensors for q_1^*, q_2^*, and q_3^* are all required. There are 6 distinct input sets that can be used to compute the 10 residuals of the complete residual set for this sensor set, and these are shown in Table 2. So, there are only $2^6 - 1 = 63$ combinations of submodels to consider, versus $2^{10-1} = 1023$ combinations of residuals. Now, if we consider only combinations of submodels that cover all the residuals, there are only 36 combinations.

So, in summary, we have at most $|Y|2^{|Y|-1}$ residuals but only at most $2^{|Y|} - 1$ submodels. So there are $2^{|Y|2^{|Y|-1}}$ combinations of residuals, versus $2^{2^{|Y|}-1}$ combinations of submodels, shrinking the search space considerably. By considering only combinations of submodels that cover all residuals, since there are at most $2^{|Y|-1}$ ways to compute each residual, there are only at most $(2^{|Y|-1})^{|Y|} = 2^{|Y|^2-|Y|}$ such combinations to consider. So we have reduced our search space from $2^{|Y|2^{|Y|-1}}$ residual combinations to $2^{2^{|Y|}-1}$ submodel combinations to $2^{|Y|^2-|Y|}$ submodel combinations that ensure there are residuals for all sensors. Clearly, this last number grows the slowest, and so we have decreased the search space over the exhaustive search algorithm by a significant factor, offering much improved scalability.

The structured search is described by Algorithm 5. An initial solution queue $\mathcal{R}$ is first constructed using single sensors. Until the initial solution queue is empty, the algorithm pops the first element off the queue, and checks if it is diagnosable. If so, it is added to a new solution set $\mathcal{R}'$, otherwise, we create new candidate solutions with one additional sensor, for each of the remaining sensors. Note here that $Y(R)$ is used to denote the sensors involved in residual set R. New candidate solutions are created only if we have not already found a diagnosable solution with smaller size (L^*). The solutions at this stage include the complete residual set for the minimum sensor sets. The purpose of this stage of the algorithm is to find the minimum sensor sets that can achieve diagnosability.

The purpose of the second stage of the algorithm is, given these minimum sensor solutions, to find optimal residual sets for each sensor set. Given one of these minimum sensor sets, we generate, for each sensor in the set, the list of potential residuals. Note here that $R_{y,Y}$ denotes the set of residuals for y that can be computed using the sensors in Y. We then use the `selectCombos` function to generate all combinations of residuals from these sets. For example, if we have two sensors y_1 and y_2 where $R_{y_1,Y} = \{r_1, r_2\}$ and $R_{y_2,Y} = \{r_3, r_4\}$, `selectCombos` would generate four residual sets: $\{r_1, r_3\}$, $\{r_1, r_4\}$, $\{r_2, r_3\}$, and $\{r_3, r_4\}$. Each of these combinations that result in diagnosability is added to a new solution set $\mathcal{R}^*$. After this loop, the best solution is picked from $\mathcal{R}^*$.

6. RESULTS

As a case study scenario, we apply the algorithms to an n-tank system with the output flows as the available sensors, and consider as the fault set all K_i^+, K_i^-, Re_i^+, Re_i^-, $Re_{i,i+1}^+$, and $Re_{i,i+1}^-$ faults. In this case, the system is only diagnosable if all the output flow sensors are included, and this should be discovered by the algorithms.

The inherent scalability of the system itself is shown in Table 3. As the number of tanks increases, the size of the complete residual set increases, as does the number of unique submodel inputs. Each tank adds a new sensor, so in the worst case, the number of unique residuals is $n2^{n-1}$. For this system, measuring the output flows allows for a substantial amount of model decomposition, so this number is reduced significantly. In fact, $|R_Y|$ increases only polynomially (third order). The number of unique U_i for the submodels grows in the worst case with $2^n - 1$, but because of the decomposition provided by the sensors, it grows only polynomially in this case (second order). Since these parameters of the tank system scale only polynomially, this cuts the worst-case search space size drastically.

Results for exhaustive search are shown in Table 4. The exhaustive algorithm finds the optimal solutions, but quickly becomes unusable due to its poor scalability. For only 4 tanks, the number of solutions that must be searched ($2^{|R_Y|} - 1$) is already over a million, and the search did not complete within a reasonable amount of time.

For 2 tanks, the optimal solution is to use only the global

Algorithm 5 $R^* = \texttt{StructuredSearch}(F, R, Y)$

$\mathcal{R} \leftarrow \varnothing$
for all $y \in Y$ **do**
  $\mathcal{R} \leftarrow \mathcal{R} \cup R_{\{y\}}$
end for
$L^* \leftarrow \inf$
$\mathcal{R}' \leftarrow \varnothing$
while $\mathcal{R}$ **not empty do**
  $R_1 \leftarrow \texttt{pop}(\mathcal{R})$
  if $\texttt{diagnosable}(F, R_1)$ **then**
    $L^* \leftarrow |R_1|$
    $\mathcal{R}' \leftarrow \mathcal{R}' \cup R_1$
  else
    if $|R_1| < L^*$ **then**
      for all $y \in Y - Y(R_1)$ **do**
        $Y' \leftarrow (Y - Y(R_1)) \cup \{y\}$
        $R_1' \leftarrow R_{Y'}$
        $\mathcal{R} \leftarrow \mathcal{R} \cup \{R_1'\}$
      end for
    end if
  end if
end while
$\mathcal{R}^* \leftarrow \{\mathcal{R}'$
for all $R_i \in \mathcal{R}'$ **do**
  $Y_i \leftarrow Y(R_i)$
  $\mathcal{R}'' \leftarrow \varnothing$
  for all $y \in Y_i$ **do**
    $\mathcal{R}'' \leftarrow R_(y, Y)$
  end for
  $\mathcal{R}'' \leftarrow \texttt{selectCombos}(\mathcal{R}'')$
  for all $R_i'' \in \mathcal{R}''$ **do**
    if $\texttt{diagnosable}(F, R_i'')$ **then**
      $\mathcal{R}^* \leftarrow \mathcal{R}^* \cup \{\mathcal{R}'\}$
    end if
  end for
end for
$R^* \leftarrow \mathcal{R}_1^*$
for all $R_i \in \mathcal{R}^*$ **do**
  if $R_i \succ R^*$ **then**
    $R^* \leftarrow R_i$
  end if
end for

model. If we use the submodel that computes q_1^* using q_2^* and the submodel that computes q_2^* using q_1^*, the system is not diagnosable, and so the global model is the optimal solution. For 3 tanks, we can improve over the global model as a solution by using two submodels computing $\{q_1^*, q_2^*\}$ and $\{q_2^*, q_3^*\}$. This decomposition is better than the global model, uses just as many sensors, and obtains complete diagnosability. We find that considering only the pressure sensors gives similar results. When considering both flow and pressure sensors, we still need one sensor for each tank, and the pressure and flow measurements can be interchanged.

The structured search algorithm attempts to avoid searching this entire space by (*i*) finding only minimum sensor sets required for diagnosability, and (*ii*) for each of these sets searching only over residual sets in which all sensors are covered. Downselecting to only residuals for a given minimum sensor set reduces the search space significantly, as does, for that given sensor set, considering only the residuals sets that cover all the sensors.

Table 3. Scalability of Tank System

Number of Tanks	Size of R_Y	Number of Unique U_i
2	4	3
3	10	6
4	20	10
5	35	15
6	56	21
7	84	28
8	120	36
9	165	45
10	220	55

Table 4. Exhaustive Search Results.

Number of Tanks	Solutions Searched	Final Solution
2	15	$(\{u_1, u_2\}, \{q_1^*, q_2^*\})$
3	1023	$(\{u_1, u_2, q_3^*\}, \{q_1^*, q_2^*\})$
		$(\{u_2, u_3, q_1^*\}, \{q_2^*, q_3^*\})$
4	1048575	N/A

Results for the structured search algorithm are shown in Table 5. Here, the number of candidates searched grows significantly slower than with the exhaustive search. Once the structured search finds a minimum sensor set from which to select residuals, the remaining search space is searched in a somewhat exhaustive way, i.e., it tries all combinations of residual sets for which the sensors are covered. Therefore, its growth is still exponential although its scalability is much improved over the exhaustive search algorithm.

Upon inspection of the solutions searched by the algorithm, we find that only a small subset are actually only worth searching. All other solutions are considering submodel sets that are not minimal. By the definition of $\succ$, as long as a solution exists using a minimal submodel set, a solution with a nonminimal submodel set will never be optimal. It is very likely that if there is a solution with a nonminimal submodel set, there is one with a minimal submodel set. If this is true, then the search space of the algorithm can be reduced even more, further improving scalability.

Table 5. Structured Search Results.

Number of Tanks	Solutions Searched	Final Solution
2	7	$(\{u_1, u_2\}, \{q_1^*, q_2^*\})$
3	34	$(\{u_1, u_2, q_3^*\}, \{q_1^*, q_2^*\})$
		$(\{u_2, u_3, q_1^*\}, \{q_2^*, q_3^*\})$
4	277	$(\{u_1, u_2, q_3^*\}, \{q_1^*, q_2^*\})$
		$(\{u_3, u_4, q_2^*\}, \{q_3^*, q_4^*\})$
5	3427	$(\{u_1, u_2, q_3^*\}, \{q_1^*, q_2^*\})$
		$(\{u_3, u_4, q_2^*, q_5^*\}, \{q_3^*, q_4^*\})$
		$(\{u_4, u_5, q_3^*\}, \{q_4^*, q_5^*\})$

Table 6. Stochastic Search Results.

Number of Tanks	Solutions Searched	Final Solution
2	1000	$(\{u_1, u_2\}, \{q_1^*, q_2^*\})$
3	1000	$(\{u_1, u_2, q_3^*\}, \{q_1^*, q_2^*\})$ $(\{u_2, u_3, q_1^*\}, \{q_2^*\})$ $(\{u_3, q_2^*\}, \{q_3^*\})$
4	1000	$(\{u_1, u_2, u_3, q_4^*\}, \{q_1^*, q_3^*\})$ $(\{u_2, u_3, u_4, q_1^*\}, \{q_2^*\})$ $(\{u_3, u_4, q_2^*\}, \{q_4^*\})$
5	1000	$(\{u_1, u_2, u_3, u_4, u_5*\}, \{q_1^*, q_5^*\})$ $(\{u_2, u_3, u_4, q_1^*, q_5^*\}, \{q_2^*, q_3^*, q_4^*\})$

Another way to increase scalability is by adding more structure to the search process, in a way that attempts to search first solutions more likely to be optimal, if diagnosable, than others. For example, we can try first the most decomposed solution (single-output submodels with the maximum number of local inputs), and then working up towards the global model. Because $\succ$ prefers more decomposed solutions, if we search candidates solutions with better decomposition first, we can terminate the search once a solution is found (since less decomposed solutions will not then be optimal). The first stage of the algorithm could also be improved by starting first with the maximum sensor set, then reducing it to find minumum subsets that still achieve diagnosability. In this case study, since all sensors are required for diagnosability, this would have resulted in finding the required sensor set much faster. In many cases it is more likely that a large subset of the sensors are needed for diagnosability rather than a small subset.

Scalability can be improved by considering heuristics to guide the search. A greedy search heuristic, for example, can improve significantly the scalability, but the resulting solutions may not be optimal.

The stochastic search algorithm is the most scalable, as the number of solutions it searches are completely defined by the user. Results for the stochastic search are given in Table 6. Here, we used $k = 10$ and $N = 100$, so 1000 candidate solutions are always searched independent of n. For 3, 4, and 5 tanks, the optimal solutions are not found. However, the solutions found are still diagnosable and represent a good model decomposition. The solutions found are nonoptimal because the submodel sets are not minimal. So, the solution space is such that there are very few optimal solutions (in this case only 1), but many good solutions. So if optimality is not a requirement, the stochastic algorithm is a suitable choice because it is likely to find a good solution since many exist in the search space. For 5 tanks, the solution is much further from optimal, so since the space is much bigger k and N should be increased.

Based on the ideas of the structured algorithm, the stochastic algorithm performance may potentially be improved. For example, it can search only a reduced space in which all sensors required for diagnosability are covered by the residual set. With a reduced space to search, with the same number of iterations it is more likely to find a better solution.

7. CONCLUSIONS

In this work, we have presented a diagnosability-based sensor placement solution by using structural model decomposition. The solution proposed in this paper analyzes the diagnosability of a system to determine the minimum set of sensors required to uniquely isolate all single faults in the system. Then, once the minimum set of sensors for complete diagnosability is computed, several criteria are taken into account to select among the set of equivalent solutions. In particular, we used three different metrics: the minimality of the involved submodel set; the number of residuals per submodel; and the total number of residuals.

In the paper we presented three different solutions for the problem. The first one, an exhaustive search, finds the optimal solutions but is not scalable. A second one, a stochastic search algorithm, sacrifices optimality for scalability. And a third one, a structured search algorithm, is more scalable than the exhaustive search while still guaranteeing optimality.

Experimental results on a multi-tank system demonstrated the performance of the algorithms and suggest possible improvements to the algorithms that will inform future work. In this paper, we considered only single faults and a continuous system for the case study, but, in future work, we will study how the to extend this solutions to multiple fault diagnosis and hybrid systems. Future work will also apply the algorithms to practical large-scale systems.

ACKNOWLEDGEMENTS

M. Daigle's and I. Roychoudhury's funding for this work was provided by the NASA System-wide Safety and Assurance Technologies (SSAT) Project. A. Bregon's funding for this work was provided by the Spanish MCI TIN2009-11326 grant.

REFERENCES

Basseville, M., Benveniste, A., Moustakides, G., & Rougee, A. (1987, Dec). Optimal sensor location for detecting changes in dynamical behavior. *IEEE Transactions on Automatic Control*, *32*(12), 1067-1075.

Bregon, A., Daigle, M., Roychoudhury, I., Biswas, G., Koutsoukos, X., & Pulido, B. (2014, May). An event-based distributed diagnosis framework using structural model decomposition. *Artificial Intelligence*, *210*, 1-35.

Casillas, M., Puig, V., Garza-Castañon, L., & Rosich, A. (2013). Optimal sensor placement for leak location in water distribution networks using genetic algorithms. *Sensors*, *13*(11), 14984–15005. doi:

10.3390/s131114984

Daigle, M. (2008). *A qualitative event-based approach to fault diagnosis of hybrid systems*. Unpublished doctoral dissertation, Vanderbilt University.

Daigle, M., Bregon, A., Biswas, G., Koutsoukos, X., & Pulido, B. (2012, August). Improving multiple fault diagnosability using possible conflicts. In *Proceedings of the 8th IFAC symposium on fault detection, supervision and safety of technical processes* (p. 144-149).

Daigle, M., Koutsoukos, X., & Biswas, G. (2007, April). Distributed diagnosis in formations of mobile robots. *IEEE Transactions on Robotics*, *23*(2), 353–369.

Debouk, R., Lafortune, S., & Teneketzis, D. (2002). On an optimization problem in sensor selection. *Discrete Event Dynamic Systems*, *12*(4), 417–445.

Eriksson, D., Krysander, M., & Frisk, E. (2012, August). Using quantitative diagnosability analysis for optimal sensor placement. Mexico City, Mexico.

Frisk, E., Krysander, M., & Åslund, J. (2009). Sensor placement for fault isolation in linear differential-algebraic systems. *Automatica*, *45*(2), 364-371.

Karnopp, D. C., Margolis, D. L., & Rosenberg, R. C. (2000). *Systems dynamics: Modeling and simulation of mechatronic systems*. New York: John Wiley & Sons, Inc.

Krysander, M., & Frisk, E. (2008). Sensor placement for fault diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, *38*(6), 1398–1410.

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

Narasimhan, S., Mosterman, P. J., & Biswas, G. (1998, May). A systematic analysis of measurement selection algorithms for fault isolation in dynamic systems. In *Proc. of the 9th international workshop on principles of diagnosis* (pp. 94–101). Cape Cod, MA USA.

Raghuraj, R., Bhushan, M., & Rengaswamy, R. (1999). Locating sensors in complex chemical plants based on fault diagnostic observability criteria. *AIChE Journal*, *45*(2), 310–322.

Rosich, A. (2012). Sensor Placement for Fault Detection and Isolation based on Structural Models. In *Proceedings of the 8th ifac symposium on fault detection, supervision and safety of technical processes, safeprocess12* (p. 391-396). Mexico City, Mexico.

Rosich, A., Frisk, E., Aslund, J., Sarrate, R., & Nejjari, F. (2012, March). Fault diagnosis based on causal computations. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *42*(2), 371-381.

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

Roychoudhury, I., Daigle, M., Bregon, A., & Pulido, B. (2013, March). A structural model decomposition framework for systems health management. In *Proceedings of the 2013 IEEE aerospace conference.*

Said, A., & Djamel, B. (2013). Optimal sensor placement for fault detection and isolation by the structural adjacency matrix. *International Journal of Physical Sciences*, *8*(6), 225–230.

Travé-Massuyès, L., Escobet, T., & Olive, X. (2006, Nov). Diagnosability analysis based on component-supported analytical redundancy relations. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *36*(6), 1146-1160.

BIOGRAPHIES

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. During the summers of 2006 and 2007, he was an intern with Mission Critical Technologies, Inc., at NASA Ames Research Center. 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, simulation, 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.

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.