Learning Diagnosis Models Using Variable-Fidelity Component Model Libraries ⋆

Alexander Feldman * **Gregory Provan** ** **Rui Abreu** * **Johan de Kleer** *

* *PARC Inc., Palo Alto, CA 94304, USA*
(e-mail: {afeldman,dekleer,rui}@parc.com)
** *Department of Computer Science, University College Cork, Cork, Ireland*
(e-mail: g.provan@cs.ucc.ie).

Abstract: System models that are used in model-based diagnosis are often composed of components drawn from component libraries. In these component libraries, there may be multiple systems of equations per component (component implementations). For example, a component may be modeled as a non-linear system (high-fidelity model), linear system, and a qualitative system (low-fidelity model). Choosing the right component model for system diagnosis is a difficult task and requires a search in the space of all possible component type combinations. In this paper we propose a method that automates this task and computes a system model that optimizes a set of diagnostic metrics in a set of diagnostic scenarios. Initial experimental results show that having linear models of some of the components in a system preserves the diagnostic accuracy and isolation time while, at the same time, improves the computational complexity and numerical stability.

1. INTRODUCTION

Model-based diagnosis (de Kleer and Williams, 1987) uses system models and sensor data to compute diagnostic hypotheses. These hypotheses have a range of applications such as decision-making (Feldman et al., 2013), repair, reconfiguration, troubleshooting, testing, and others. While providing many benefits, model-based diagnosis is expensive due to the need to obtain/construct good system models. To amortize this high modeling cost, researchers develop and use component libraries.

A component may have several representations in a component library. For example, a NAND-gate may be modeled as a system of non-linear equations that govern the analogue electrical laws of the gate, or its linear approximation or with the simple Boolean expression $o \leftrightarrow \neg(i_1 \wedge i_2)$. Although one may postulate that the best choice for a component model is the one that most accurately represents the physics (in the case of the NAND-gate, this would be the analogue electrical model), experiments show that the result of this choice is sometimes hard to predict and is dependent on the diagnostic context.

In this article we propose the novel approach of performing diagnostics inference using component model libraries consisting of multiple models of differing fidelity. We present an approach to automatically compose a global diagnosis model from different types of sub-models from a library, such that the model maximizes a given score metric over a set of diagnosis test cases. We use a test-set of diagnostic scenarios for learning the optimal system model, where the test-set is artificially generated (e.g., by simulation) and contains a representative set of likely faults. The algorithm we propose chooses component models that optimize some (weighted) diagnostic metrics such as diagnostic accuracy (which is dual of classification errors), isolation time, or computational complexity (Feldman et al., 2010). The output of the algorithm is a system model that can be later used for on-line diagnosis.

⋆ Supported by SFI grant 12/RC/2289.

To illustrate the usability of our algorithm, consider a diagnostic model of a crane. The model would contain parts such as electrical motors and drives and a Programmable Logic Controller (PLC). Non-linear electromechanical-models are most appropriate choice for the moving parts, however modeling the PLC with non-linear equations would result in a suboptimal diagnosis due to high complexity. Further, high simulation accuracy does not necessarily translate to high diagnostic accuracy. The algorithm we propose would run a few diagnostic scenarios and then would discard the high-fidelity PLC model and instead use the computationally simpler Boolean/qualitative/state-machine components. By doing this, the algorithm optimizes the diagnostic performance of an MBD solver on a set of diagnostic test cases.

Our contributions are as follows. We describe an algorithm for learning models with components of variable fidelity. We illustrate our algorithm on a multiple-tank dynamic system benchmark, consisting of multiple tanks connected with valves. We show that the algorithm that we propose is capable of designing models that result in non-intuitive trade-offs during diagnosis.

2. RELATED WORK

This work bears some relation to the use of ensembles in learning. An ensemble method is a supervised learning-based approach to discover which combination of model has the best predictive power (Kuncheva and Whitaker, 2003). Model ensemble methods have been applied in disciplines ranging from statistics to AI (e.g., (Breiman, 1996; Wolpert, 1992)), and it has been shown that an ensemble is often more accurate than any single model in the ensemble (Maclin and Opitz, 2011).

The motivation behind using a model ensemble is to build a predictive model by integrating multiple models to achieve better performance than could be obtained from using models individually (Maclin and Opitz, 2011; Rokach, 2010). The

most popular ensemble algorithms are Bagging and Boosting, which are meta-algorithms that pool decisions from multiple classifiers. Both these algorithms apply the Mixture of Experts paradigm (Brown, 2010).

In the context of software fault localization using spectrum-based fault localization (Abreu et al., 2007), approaches have been proposed to combine several heuristics. In (Wang et al., 2011) a search-based algorithm to combine the existing heuristics. Xuan et al. (2014) proposed a machine learning-based approach to combine multiple diagnostic rankings metrics. An approach based in data fusion to heuristic combination was proposed in (Lo et al., 2014).

Another body of related work is research related to choosing the right level of modeling abstraction. As an example, in the model-based diagnosis literature there has been considerable work on diagnostic assumptions and selecting appropriate models for a diagnostic task (Struss, 1992). The work proposed in (de Kleer, 2007) focuses primarily on assumptions associated with choosing domain abstractions.

There has been considerable research on structural abstraction (Chittaro and Ranon, 2004; Hamscher, 1990) where groups of components are combined to form larger systems to reduce computational complexity. The work in (Sachenbacher and Struss, 2005) describes how the task can be used to partition the value of a variable into the qualitative values needed to solve a task. In (Torta and Torasso, 2003) presents another approach to partition the value of a variable into qualitative ranges to reduce complexity when there is limited observability of the variables.

3. RUNNING EXAMPLE

We illustrate our concepts using a three-tank system, as shown in figure 1. The tanks are denoted as T_1, T_2, and T_3. They all have the same area $A_1 = A_2 = A_3 = 3$ [m^2]. The three tanks are indestructible and of infinite height representing an idealized experiment. We assume that $g = 10$ and the liquid is "pure" water with density $\rho = 1$.

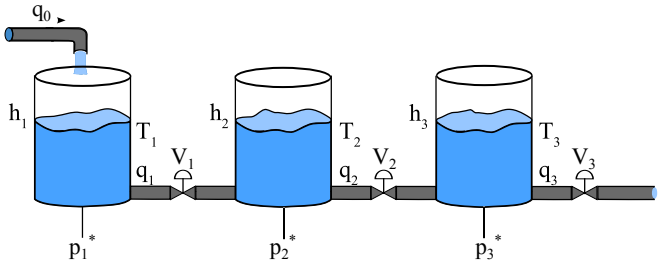

Fig. 1. Diagram of the three-tank system.

Tank T_1 is filled from a pipe q_0 with a constant flow of 0.75 [m^3/s]. It drains into T_2 via a pipe q_1. The liquid level is denoted as h_1. There is a pressure sensor p_1 connected to T_1 that measures the pressure in Pascals [Pa]. Starting from Newton's (and Bernouli's) equations and manipulating them (the actual derivation is irrelevant in this paper) we derive the following Ordinary Differential Equation (ODE) that gives the level of the liquid in T_1:

$$\frac{dh_1}{dt} = \frac{q_0 - k_1\sqrt{h_1 - h_2}}{A_1} \tag{1}$$

In eq. 1, the coefficient k_1 is the product of the cross-sectional area of the tank A_1 and the area of the drainage hole and $\sqrt{2g}$ and the friction/contraction factor of the hole. We emphasize the use of k_1 because, later, we will be "diagnosing" our system in term of changes in k_1. Consider a physical valve R_1 between T_1 and T_2 that constraints the flow between the two tanks. We can say that the valve changes proportionally to the cross-sectional drainage area of q_1 and hence k_1. The diagnostic task is to compute the true value of k_1, given p_1, and from k_1 we can compute the actual position of the valve R_1. The water levels of T_2 and T_3, denoted as h_2 and h_3 respectively, are given by:

$$\frac{dh_i}{dt} = \frac{k_{i-1}\sqrt{h_{i-1} - h_i} - k_i\sqrt{h_i}}{A_i}, \tag{2}$$

where i is the tank index ($i \in \{2, 3\}$).

We assume that $k_1 = k_2 = k_3 = 0.75$.

Finally, we turn the water level into pressure:

$$p_i = \frac{g\, h_i\, A}{A} = g\, h_i \tag{3}$$

where i is the tank index ($i \in \{1, 2, 3\}$).

We assume that the initial water level in the three tanks is zero.

4. CONCEPTS AND DEFINITIONS

All concepts, definitions, and algorithms discussed in this article are illustrated on a commonly used dynamic system consisting of water tanks that are connected with valves.

4.1 Objective

We present an approach to automatically compose a global diagnosis model Φ from a library $\mathcal{C}$ of different types of sub-models, such that the model maximizes a given scoring function Γ over a set $\mathcal{A}$ of diagnosis test cases.

We formalise our objective as optimizing a model such that:

$$\phi^* = \underset{\phi_i \in \mathbf{\Phi}}{\operatorname{argmin}}\, \Gamma\left(\mathcal{A}, \mathcal{C}, \phi_i\right). \tag{4}$$

where $\mathbf{\Phi}$ is the space of all models composable from $\mathcal{C}$ according to a model specification denoted by Φ.

In the following, we specify our notions of model library, model specification, and scoring function.

4.2 Compositional Modeling

We define a component library $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ to consist of a set of component types, such that type C_i, for $i = \{1, 2, \ldots, n\}$ is defined as *a set* of models. Each type C_i contains non-linear, linear, and qualitative models describing the behavior of C_i.

A decomposable model can be described using two orthogonal aspects: *behavior* and *topology* (interaction). The behavior model describes the (possibly dynamic) behaviors of the system and components. The topology model describes component connectivity in terms of components and their connections, and defines the constraints on component behaviors that enable their interactions to be specified at the system-level.

Definition 1. (Topology). We describe the system topology of a composable system using a graph $G(V, E)$, where vertices V correspond to components and edges in E correspond to connections between components.

Given the system topology and equations from the component models, we can then specify a system description as follows.

Definition 2. (System Description). Given a component library $\mathcal{C}$, a topology $G(V, E)$, and some law of composition $\mathcal{L}$, a system description $\text{SD} = \langle \Phi, \text{COMPS}, \text{OBS} \rangle$ is defined as a set of equations Φ, a set of component variables COMPS, and a set of observable variables OBS.

Let us continue with some notation and definitions.

4.3 Model Library Classes

Consider a component library $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ where each component C_i for $i = \{1, 2, \ldots, n\}$ is defined as *a set* of models. Each set $C_i = \{C_{i,1}, C_{i,2}, C_{i,3}\}$ contains either a non-linear, linear, or qualitative model, which describes the behavior of component C_i.

We consider systems that can be described in terms of a set $\boldsymbol{x}(t)$ of state variables, $\boldsymbol{y}(t)$ of observable variables, and $\boldsymbol{u}(t)$ of control variables.

Definition 3. (Non-Linear Model). We write the dynamic equations for a model in state-space form using

$$\dot{\boldsymbol{x}}(t) = \psi(\boldsymbol{x}(t)) + \boldsymbol{u}(t)) \tag{5}$$

$$\boldsymbol{y}(t) = \gamma(\boldsymbol{x}(t)), \boldsymbol{u}(t)), \tag{6}$$

where ψ and γ are non-linear functions.

In the case of the three-tanks example, the non-linear model of T_1 is given by equations (1) and (3) and the models of T_2 and T_3 are given by equations (2) and (3). We also include in our model library a class of linear models, which are derived from the non-linear models via model linearization Spanos (1977), e.g., mapping the system behaviour around an equilibrium point Roubal et al. (2010), or replacement of non-linear with linear operators.

Definition 4. (Linear Model). We write the linear dynamical equations for a model in state-space form using

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t) + \mathbf{C}\boldsymbol{\omega}(t) \tag{7}$$

$$\boldsymbol{y}(t) = \mathbf{D}(\boldsymbol{x}(t)), \tag{8}$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are linear matrices, and $\boldsymbol{\omega}(t)$ is a fault vector.

For the linear three-tank model we replace the non-linear sub-function $\sqrt{h_{i-1} - h_i}$ with the linear sub-function $\gamma_i(h_{i-1} - h_i)$, where γ_i is a parameter (to be estimated) governing the flow between tanks $i-1$ and i. We obtain the following system equations for tanks T_2 and T_3:

$$\frac{dh_i}{dt} = \frac{k_{i-1}(h_{i-1} - h_i) - k_i h_i}{A_i}, \tag{9}$$

Definition 5. (Qualitative Model). We write the dynamical equations for a qualitative model in state-space form using

$$\dot{\boldsymbol{x}}(t) = \upsilon(\boldsymbol{x}(t)) + \boldsymbol{u}(t)) \tag{10}$$

$$\boldsymbol{y}(t) = \mu(\boldsymbol{x}(t)), \boldsymbol{u}(t)), \tag{11}$$

where υ and μ are functions from the set of reasonable functions f such that $f' > 0$ on the interior of its domain (Kuipers and Åström, 1994).

For the qualitative model we replace the non-linear sub-function $\sqrt{h_{i-1} - h_i}$ with the qualitative $M^+(h_{i-1} - h_i)$, where M^+ is the set of reasonable functions f such that $f' > 0$ on the interior of its domain (Kuipers and Åström, 1994).

$$\frac{dh_i}{dt} = \frac{1}{A_2}\left[\kappa_1 M^+(h_i - h_{i-1}) - \kappa_2 M^+(h_i)\right] \tag{12}$$

The tank heights are constrained to be non-negative. As a consequence, we can discretize the h_i to take on values $\{+, 0\}$, which means that $M^+(h_i)$ can take on values $\{+, 0, -\}$. The domain for $\frac{dh_1}{dt}$ must be $\{+, 0, -\}$, since q_0 is non-negative and each $M^+(h_i - h_j)$ can take on values $\{+, 0, -\}$.

4.4 Diagnostic Problem

This section describes our notion of diagnostics problem.

Definition 6. (Observation). Given a system description SD, an observation $\tilde{\alpha} = \langle \alpha, t_{\text{obs}} \rangle$ is an instantiation of the variables in OBS at a time instant t_{obs}.

One possible observation for our running example is: $p_1 = 142.4$, $p_2 = 26.8$, and $p_3 = 13$ at $t_{\text{obs}} = 300$.

Definition 7. (Fault Injection). Given a system description SD, a fault injection $\tilde{\varepsilon} = \langle \varepsilon, t_{\text{inj}} \rangle$ is an instantiation of the variables in COMPS at a time instant t_{inj}.

For the three-tanks example, fault injection values of $R_1 = 0.5$ at time $t_{\text{inj}} = 250$ would correspond to the first valve being stuck at 50%.

Definition 8. (Diagnosis). Given a system description SD, a diagnosis $\tilde{\omega} = \langle \omega, t_{\text{diag}} \rangle$ is a probabilistic assignment of the variables in COMPS at a time instant t_{diag}.

Continuing with the example, a diagnosis that reflects the given observation and non-linear model of the three tanks is $\Pr(R_1 = 0.5) = 0.931)$ at time $t_{\text{diag}} = 310$ which isolates the fault in 60 s with high accuracy.

All the above definitions are used in formulating the main diagnostic problem for a dynamic system:

Definition 9. (Diagnostic Problem). A diagnostic problem DP is defined as the quadruple $\text{DP} = \langle \text{SD}, \tilde{\alpha}, \tilde{\varepsilon}, \tilde{\omega} \rangle$.

4.5 Diagnostic Performance Metrics

Unlike other AI disciplines, in MBD there are multiple factors that should be considered when applying performance metrics to real-world systems. We define a metric vector $\Gamma = \{\gamma_1, ..., \gamma_m\}$ to represent the set of relevant metrics. The most important computational metric is the number of diagnostic errors, which is dual to the isolation accuracy (Feldman et al., 2010).

Definition 10. (Diagnostic Errors). Given a diagnostic problem DP the diagnostic errors metric M_{err} is defined as:

$$\gamma_1 = M_{\text{err}} = \sum_{c \in \text{COMPS}} |\Pr(\omega_c \neq \text{OK}) - \text{Inj}(\varepsilon_c)| \tag{13}$$

The second most important metric for a dynamic system is the time between a fault is injected and when the algorithm detects a fault.

Definition 11. (Isolation Time). Given a diagnostic problem DP the isolation time metric M_{iso} is defined as:

$$\gamma_2 = M_{\text{iso}} = t_{\text{diag}} - t_{\text{inj}} \tag{14}$$

Diagnostic algorithms are typically given a system model SD and a set of test cases $\mathcal{A} = \{\langle\tilde{\alpha}_1, \tilde{\varepsilon}_1\rangle, \langle\tilde{\alpha}_2, \tilde{\varepsilon}_2\rangle, \cdots, \langle\tilde{\alpha}_n, \tilde{\varepsilon}_n\rangle\}$. The main goal of these diagnostic algorithms is to optimize a superposition of the diagnostic metrics. Each diagnostic metric is weighted with a domain specific coefficient (these are $g_1 = g_{\text{err}}$, and $g_2 = g_{\text{iso}}$, respectively, in the the cases of $\gamma_1 = M_{\text{err}}$, and $\gamma_2 = M_{\text{iso}}$). In this article, however, we solve an orthogonal problem: given a diagnostic algorithm, a component library and a set of test cases $\mathcal{A}$, compute a model composition SD such that $g_1\gamma_1 + g_2\gamma_2$ is minimized.

5. VARIABLE-FIDELITY MODEL LEARNING ALGORITHM

This section describes our learning algorithm for model generation, shown in algorithm 1. This is a high-level description, and the algorithm uses a diagnostic engine similar to the one described by Feldman et al. (2013).

Algorithm 1: COMPOSEMODEL($G, \mathcal{C}, \mathcal{A}$)

Input: G, model topology
Input: $\mathcal{C}$, component library
Input: $\mathcal{A}$, set of test scenarios
Result: SD, model
Local variables: SD*, SD$'$, models, initially $\emptyset$
Local variable: α, test scenario
Local variable: ω, diagnosis
Local variable: γ, diagnosis score
Local variable: $\gamma_{\min}$, optimal diagnosis score, initially ∞

```
repeat
    SD* ← NEXTMODELCOMPOSITION(G, C, SD')
    SD' ← SD*
    m ← 0
    foreach α ∈ A do
        ω ← DIAGNOSE(SD*, α)
        γ ← γ + EVALUATE(α, ω)
    if γ < γ_min then
        γ_min ← γ
        SD ← SD*
until TERMINATE?(SD*, SD', m);
return SD
```

Algorithm 1 is non-deterministic. The non-determinism is in the auxiliary function NEXTMODELCOMPOSITION (line 1). This function takes a model topology and a component library as inputs and returns a composed model. Each component has multiple representations in the component library (e.g., qualitative, linear, non-linear). The total number of component combinations is $O(n^{|\text{COMPS}|})$, where n is the number of representations. Fortunately, there is no need to perform a complete search over the space of all possible model compositions. A greedy-search strategy achieves satisfactory performance in most practical cases.

The subroutine DIAGNOSE in line 1 implements a diagnostic oracle. Given a system description SD*, and an observation α, it computes a diagnosis ω (see definition 8). This diagnosis can be compared to the fault injection to compute one or more diagnostic metrics. This is done by the EVALUATE subroutine in line 1. The combined metric result is accumulated in a variable m. Algorithm 1 assumes that larger metrics are worse, e.g. metric results are penalties, and it chooses the model that minimizes the cumulative penalty. The assumption is that the set of test scenarios is representative and the learned model SD is going to minimize future scenarios with unknown faults.

Consider the three-tanks example. The subroutine NEXTMODELCOMPOSITION first generates a model where T_1 is non-linear, T_2 is non-linear, and T_3 is non-linear. In the second call NEXTMODELCOMPOSITION changes T_1 from non-linear to linear. Depending on the implementation of NEXTMODELCOMPOSITION the next candidate can be either T_1 and T_2 non-linear while T_3 linear or T_1 non-linear, T_2 linear, and T_3 non-linear.

In our implementation we provide the following search policies:

Breadth-First Search (BFS): This search policy starts with all components having the same model types (for example non-linear), then considers all models with a single component type change. After all single component type changes are exhausted, the algorithm continues with pairs of components, then triples, etc.

Depth-First Search (DFS): The algorithm starts by changing the type of the first component, then the second, etc., until all component types are changed. At this point, algorithm 1 backtracks one step, generates a sibling assignment and continues traversing down and backtracking in the same manner until no more backtracking is possible.

Forward Greedy Stochastic Search (FGSS): This is a randomized search policy. In this mode, the algorithm starts by changing the type of one of the components. If the change improves the metric in line 1 of algorithm 1, then the change is accepted (see lines 1–1 of algorithm 1). This is our preferred search policy as typically the evaluation metric improves monotonically when changing the component types one by one.

Backwards Greedy Stochastic Search (BGSS): In this mode, the search starts with all component types changed from their defaults. The type of a random component is then flipped and the flip is retained iff the flip leads to a decrease in the total metric evaluation score. The order of components is arbitrary. As the whole search process is stochastic, it needs to be run multiple iterations are necessary in order to achieve the desired completeness.

The computational performance of algorithm 1 is determined by the search choice, given an efficient implementation of the DIAGNOSE subroutine:

Proposition 1. Algorithm 1 will terminate in $O(m^n)$ calls to a diagnostic oracle, given exhaustive search (BFS and DFS) and in $O(n)$ calls to a diagnostic oracle given a greedy search (FGSS or BGSS).

The proof of proposition 1 is elementary from the number of combinations of component type in each component ensemble.

Like any learning method, algorithm 1 assumes that the future is predictable given a test set of diagnostic scenarios. The model computed by algorithm 1 will optimize the cumulative metrics

for this past set of scenarios but there is, of course, no guarantee that the model is optimal in the general case.

6. EMPIRICAL ANALYSIS

This section discusses empirical results from various tank models. We adopt as our "gold standard" model a model of the highest fidelity, i.e., a model composed entirely of non-linear components.

Table 1 shows a benchmark consisting of variations of the 3-tank model. For each system, we have experimented with a single and with a double-fault injected at 250 and 275 seconds. We have evaluated the diagnostic performance according to three diagnostic metrics: (1) the diagnostic accuracy M_{err}, (2) the isolation time t_{iso}, and (3) the CPU time t_{cpu}.

		All tanks non-linear			First tank linear		
Number of tanks	Fault card.	M_{err}	t_{iso}	t_{cpu}	M_{err}	t_{iso}	t_{cpu}
2	1	241.8	3	10.59	997	1	8.52
2	2	807.2	3	10.64	1214.7	1	8.56
3	1	241.8	3	23.7	1075.6	1	17.96
3	2	805.6	3	24.04	1243.3	1	18
4	1	241.8	3	39.47	1191.9	1	34.59
4	2	813.7	3	39.5	1636.2	1	34.51

Table 1. Diagnostic quality for a benchmark of ensembles

The results in table 1 show that the automatic model composition that we propose in this paper can lead to useful results. In the partly-linearized model, although we have an increased number of classification error, we benefit with shorter isolation time and smaller number of CPU cycles. These effects are not very well expressed due to the nature of the benchmark. We expect significantly larger savings with a hybrid system: for example one having logic gates best modeled by Boolean equations and real-valued variables. We next look at these experiments in detail.

Figure 2 shows the results from one test case: a failure injection (valve R_1 stuck at 50%) at $t = 250$. For this simulation we use the highly accurate non-linear model. The results from this simulation show that at the time of the fault injection, the water level in tank T_1 starts increasing while the water level at tanks T_2 and T_3 start decreasing due to the lower inflow.

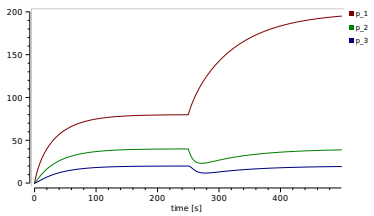

Fig. 2. Fault injection at $t = 250$ [s]

Next we use the simulation output shown in figure 1 to create a test case for algorithm 1. The result of performing diagnosis with the same non-linear model (that was used in simulation) are shown in figure 3. We can see that the diagnostic accuracy is high (the number of classification errors at $t = 432$ [s] is

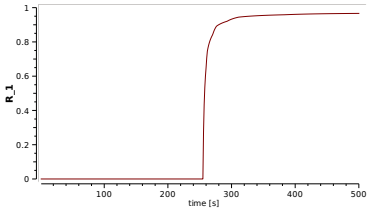

Fig. 3. Probability of R_1 being at fault with all components non-linear

only $M_{\text{err}} = 0.034$). The isolation accuracy is also very high ($M_{\text{iso}} = 7$ [s]).

Figure 4 shows the diagnostic accuracy and isolation time with an all-linear model. This all-linear model delivers both poor diagnostic accuracy (classification errors) and poor isolation time. After the fault injection at $t = 250$ [s], the probability estimation is better and the faulty valve appears at the top of the list of faulty components. The benefit of the all-linear model is that it is much faster to simulate.

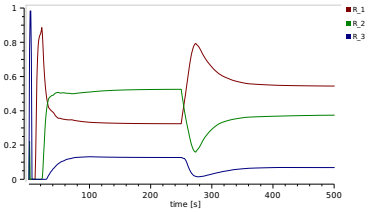

Fig. 4. Probability of R_1, R_2, and R_3 being at fault with all components linear

The plot in figure 5 shows the diagnostic performance with partly linear, partly non-linear model (T_1 is non-linear, while T_2 and T_3 are linear). The diagnostic accuracy is almost the same as the gold standard model, except a false-positive detection at the beginning of the scenario that can be ignored. Of course, the number of computations for simulating the partly linear model is significantly smaller (recall that we need multiple simulations for diagnosis). As a result this model is a preferred compromise for diagnostic accuracy and computational complexity.

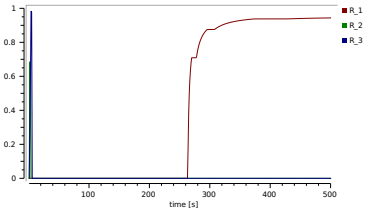

Fig. 5. Probability of R_1, R_2, and R_3 being at fault with T_1 non-linear and both T_2 and T_3 linear

Algorithm 1 is insensitive to the cardinality of the fault injection. Figure 6, for example, shows the result of a double-fault injection. The first fault represents a 50 %-stuck valve R_1 at $t = 250$ and the second fault represents a stuck-valve R_2 at 75 % at $t = 250$. The figure shows that the effect of the second fault is less pronounced due to its magnitude.

Figure 7 shows the fault-probability for the double-fault injection shown in figure 6 with a fully non-linear model. For this, we have configured the diagnostic reasoner to consider double-fault hypotheses (see Feldman et al. (2013)). Both faults

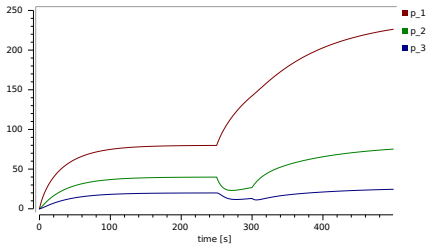

Fig. 6. Fault injection of a double fault at $t = 250$ [s] and $t = 300$ [s]

are captured with short isolation time. Classification error is $M_{\text{err}} = 0$ and $M_{\text{iso}} = 9$ while for the second (cumulative) fault we have $M_{\text{err}} = 0.3545$ and $M_{\text{iso}} = 31$.

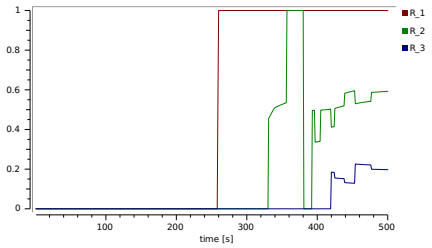

Fig. 7. Probability of finding the double-fault with all components non-linear

The results from our experiments are non-intuitive in a sense that if we replace a non-linear component model with a linear one it does not affect the diagnostic accuracy while it does improve other metrics. This cannot be done mechanically for all components as the diagnostic accuracy may deteriorate suddenly. Our results indicate that the role of the topology for diagnostic reasoning is important.

7. CONCLUSIONS

Despite successful applications within the machine learning community (Brown, 2010; Dietterich, 2000), model ensembles have not been adopted in diagnostics inference. This paper addresses that gap by proposing an algorithm that, given a test set of diagnostic scenarios, learns the optimal system model from a library of component models.

We have shown the proposed algorithm on a dynamic system consisting of three tanks connected with valves. Results indicate that the model composition may be non-intuitive and suggest that the choice of modeling abstraction depends on the model topology and cannot be preconceived during the design of the component library.

Future work includes a thorough validation of the proposed algorithm using different topologies, modeling abstraction libraries, and different fault scenarios. We are also interested in investigating the impact of the learning phase in the detection phase of the algorithm.

REFERENCES

Abreu, R., Zoeteweij, P., and Van Gemund, A.J. (2007). On the accuracy of spectrum-based fault localization. In *Testing: Academic and Industrial Conference Practice and Research Techniques*, 89–98. IEEE.

Breiman, L. (1996). Stacked regressions. *Machine learning*, 24(1), 49–64.

Brown, G. (2010). Ensemble learning. In *Encyclopedia of Machine Learning*, 312–320. Springer.

Chittaro, L. and Ranon, R. (2004). Hierarchical model-based diagnosis based on structural abstraction. *Artificial Intelligence*, 155(1), 147–182.

de Kleer, J. (2007). Dynamic domain abstraction through meta-diagnosis. In *Abstraction, Reformulation, and Approximation*, 109–123. Springer.

de Kleer, J. and Williams, B. (1987). Diagnosing multiple faults. *Artificial Intelligence*, 32(1), 97–130.

Dietterich, T.G. (2000). Ensemble methods in machine learning. In *Multiple classifier systems*, 1–15. Springer.

Feldman, A., de Castro, H.V., van Gemund, A., and Provan, G. (2013). Model-based diagnostic decision-support system for satellites. In *Proceedings of the IEEE Aerospace Conference, Big Sky, Montana, USA*, 1–14.

Feldman, A., Kurtoglu, T., Narasimhan, S., Poll, S., Garcia, D., de Kleer, J., Kuhn, L., and van Gemund, A. (2010). Empirical evaluation of diagnostic algorithm performance using a generic framework. *International Journal of Prognostics and Health Management*, 1–28.

Hamscher, W. (1990). Xde: Diagnosing devices with hierarchic structure and known component failure modes. In *Artificial Intelligence Applications, 1990., Sixth Conference on*, 48–54. IEEE.

Kuipers, B. and Åström, K. (1994). The composition and validation of heterogeneous control laws. *Automatica*, 30(2), 233–249.

Kuncheva, L.I. and Whitaker, C.J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2), 181–207.

Lo, D., Xia, X., et al. (2014). Fusion fault localizers. In *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering*, 127–138. ACM.

Maclin, R. and Opitz, D. (2011). Popular ensemble methods: An empirical study. *arXiv preprint arXiv:1106.0257*.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2), 1–39.

Roubal, J., Husek, P., and Stecha, J. (2010). Linearization: Students forget the operating point. *Education, IEEE Transactions on*, 53(3), 413–418.

Sachenbacher, M. and Struss, P. (2005). Task-dependent qualitative domain abstraction. *Artificial Intelligence*, 162(1), 121–143.

Spanos, P.D. (1977). *Linearization techniques for non-linear dynamical systems*. Ph.D. thesis, California Institute of Technology.

Struss, P. (1992). Whatś in SD? towards a theory of modeling for diagnosis. *Readings in model-based diagnosis*, 419–449.

Torta, G. and Torasso, P. (2003). Automatic abstraction in component-based diagnosis driven by system observability. In *IJCAI*, 394–402.

Wang, S., Lo, D., Jiang, L., Lau, H.C., et al. (2011). Search-based fault localization. In *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering*, 556–559. IEEE Computer Society.

Wolpert, D.H. (1992). Stacked generalization. *Neural networks*, 5(2), 241–259.

Xuan, J., Monperrus, M., et al. (2014). Learning to combine multiple ranking metrics for fault localization. In *ICSME-30th International Conference on Software Maintenance and Evolution*.