Qualitative Event-Based Fault Isolation under Uncertain Observations

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

For many systems, automatic fault diagnosis is critical to ensuring safe and efficient operation. Fault isolation is performed by analyzing measured signals from the system, and reasoning over the system behavior to determine which faults have occurred, based on models of predicted faulty behavior. For dynamic systems, reasoning may be performed using qualitative analysis of the differences between measured signals and their predicted values, in which observations take the form of qualitative symbols. Such an approach is quick to isolate faults, but depends critically on correct generation of the qualitative symbols from the signals. In this paper, we develop an approach to qualitative event-based fault isolation for dynamic systems that is robust to incorrect qualitative observations. Observations are treated as uncertain, where multiple interpretations of an observation, each with its own probability, are considered. By interpreting observed symbols in a probabilistic manner, the approach degrades gracefully as the number of incorrectly-generated symbols increases. The approach is demonstrated on an electrical power system testbed, and experiments using real data obtained from the hardware demonstrate the improved fault isolation performance in the presence of incorrect symbol generation.

1. INTRODUCTION

For many systems, automatic fault diagnosis is critical to ensuring safe and efficient operation. Within fault diagnosis, the task of fault isolation is concerned with an analysis of observed behavior in order to determine which fault has occurred. In many approaches, observations are transformed into a discrete symbolic (e.g., qualitative) form over which reasoning can be performed (Puig, Quevedo, Escobet, & Pulido, 2005; Koscielny & Zakroczymski, 2000). For dynamic systems, these discrete observations take the form of events (Daigle, Koutsoukos, & Biswas, 2009).

In qualitative fault isolation, residual signals are computed as the differences of observed behavior and predicted nominal behavior (Mosterman & Biswas, 1999). Deviations of the residual signals are then abstracted into symbolic, qualitative representations, called fault signatures, to facilitate diagnostic reasoning (specifically, +, −, and 0 symbols, representing increase, decrease, and no change from nominal, respectively). Fault models describe the potential sequences of fault signatures produced by faults, forming a qualitative event-based fault isolation approach (Daigle et al., 2009). Such an approach is quick to isolate faults, but depends critically on correct generation of these qualitative fault signatures. When the transformation from observed quantitative signals into observed qualitative fault signatures does not produce the correct result, the wrong information will be used to isolate faults, and this incorrect signature generation will, therefore, lead to incorrect diagnoses.

In this paper, we develop an *observation-robust* approach to qualitative event-based fault isolation for dynamic systems as an extension and generalization of the approach in (Daigle et al., 2009). Here, observation-robust means that the approach is still successful, to some degree, when encountering incorrect observations (henceforth, by *observation* we mean the version of the quantitative signal transformed into a qualitative symbol). By considering the qualitative observations as uncertain, and interpreting them in a probabilistic manner, the approach degrades gracefully as the number of incorrectly-generated symbols increases. The approach is

demonstrated on the Advanced Diagnostics and Prognostics Testbed (ADAPT) (Poll et al., 2007) an electrical power system testbed that has served as a benchmark diagnostic system in the diagnostics community (Poll et al., 2011; Sweet, Feldman, Narasimhan, Daigle, & Poll, 2013). Using real experimental data obtained from the ADAPT hardware, we demonstrate the improved fault isolation performance in the presence of incorrect symbol generation.

Several previous works have used probabilistic solutions for different tasks of the fault diagnosis problem. In (Ricks & Mengshoel, 2009) the authors use Bayesian Networks (BNs) to represent probabilistic multi-variate models, which are applied to the ADAPT hardware, as we do in this paper. Other works have also applied BNs or Dynamic BNs (DBNs) for fault diagnosis, e.g., in (Pernestål, 2009) the author uses DBNs to improve the diagnosis of automotive vehicles, and in (Alonso-Gonzalez, Moya, & Biswas, 2011; Roychoudhury, 2009; Roychoudhury, Biswas, & Koutsoukos, 2010) DBNs are used for fault diagnosis. In all these cases, the probabilistic solutions are used to model the systems under conditions of uncertainty and then to perform diagnosis. However, more sources of uncertainty appear in the fault diagnosis process due to, for example, improper threshold selections or incorrect symbol generation. Our approach in this paper uses a model based on physical equations of the system, and performs fault diagnosis using this model. The probabilistic methods are then used to reduce the uncertainty in fault isolation due to incorrectly-generated symbols. An approach similar to our work is presented in (Ying, Kirubarajan, Pattipati, & Patterson-Hine, 2000), in the sense that a probabilistic solution is used to perform fault diagnosis in systems with imperfect diagnosis tests. However, the diagnosis approach and the probabilistic solution are different than those used in this paper.

The remainder of the paper is organized as follows. Section 2 formulates the problem for event-based fault isolation. Section 3 reviews the standard event-based fault isolation approach, and Section 4 extends the approach to be observation-robust. Section 5 describes implementations of the standard and robust frameworks based on qualitative fault isolation, and presents the case study and results. Section 6 concludes the paper and discusses future work.

2. PROBLEM FORMULATION

In this section, we define the fault isolation problem that we aim to solve. We assume an event-based fault isolation framework, where faults are isolated based on the analysis of a sequence of observable events produced as a result of the fault occurrence (where, in the nominal case, no such events are produced). The approach is related to discrete-event diagnosis (Sampath, Sengupta, Lafortune, Sinnamohideen, & Teneketzis, 1996) and, more closely, the concept of chronicles (Cordier & Dousson, 2000). For the purposes of defining the problem and describing the fault isolation approach, we present a generalized theoretical framework for event-based fault isolation. In Section 5, we will describe a specific implementation of this framework for dynamic systems (Daigle et al., 2009).

First, we have the set of faults, F, that may occur in the system. Faults produce observable events, called *fault signatures*.

Definition 1 (Fault Signature)**.** A *fault signature* for a fault f denoted by σ_f, is an event that is observed as a consequence of the occurrence of f. The set of fault signatures for f is denoted as Σ_f. The set of fault signatures over a set of faults F is denoted as Σ_F, i.e., $\Sigma_F = \bigcup_{f \in F} \Sigma_f$.

These events are produced in some temporal order. A *fault trace* is a one particular fault signature sequence that may be observed.

Definition 2 (Fault Trace)**.** A *fault trace* for a fault f denoted by λ_f, is a sequence of fault signatures from Σ_f resulting from the occurrence of f.

Definition 3 (Maximal Fault Trace)**.** A fault trace λ_f for a fault f is *maximal* if there is no extension $\lambda_f \sigma_f$ that is also a fault trace for f.

The set of all possible maximal fault traces for a fault is called its *fault language*.

Definition 4 (Fault Language)**.** The *fault language* of a fault $f \in F$ denoted by L_f, is the set of all maximal fault traces for f. The union of fault languages for a set of faults F is denoted as L_F, i.e., $L_F = \bigcup_{f \in F} L_f$.

We assume that we have considered all possible faults in F, and that the fault languages are complete.

Assumption 1 (Completeness of F)**.** We assume that F is complete, i.e., there is no other fault $f \notin F$ that can occur.

Assumption 2 (Completeness of L_f)**.** We assume that for every fault $f \in F$, L_f is complete, i.e., there is no other maximal fault trace $\lambda_f \notin L_f$ that may occur as a result of f.

By Assumptions 1 and 2, whenever some fault trace λ occurs, it must have been produced by some fault $f \in F$, and it must belong to L_f for at least one $f \in F$. These assumptions are quite standard in model-based diagnosis. In some approaches, e.g., (Hofbaur & Williams, 2002; Narasimhan & Brownston, 2007), an *unknown fault* is considered, which is consistent with everything. In our approach, such a fault could be included by adding a new f where L_f contains all possible traces.

So, associated with each fault is a set of fault traces, where the maximal fault traces are collected into a fault language. When a fault occurs, a specific event sequence will be observed that belongs to the fault language. In this framework,

Algorithm 1 $F^* \leftarrow$ `FaultIsolation`(F)

1: $F^* \leftarrow F$
2: $\lambda \leftarrow \varnothing$
3: **while** σ_i **observed do**
4: $\quad \lambda \leftarrow \lambda\sigma_i$
5: $\quad F^* \leftarrow$ `FindConsistentFaults`(F^*, λ)
6: **end while**

fault isolation reduces to matching observed fault traces to predicted fault traces, to determine which fault has occurred. So, the fault isolation problem is defined as follows.

Problem. Given an observed fault trace, λ, find the *most likely* single fault f that produced λ.

Here, we aim to find the *most likely* fault, because the observed fault trace may not always be generated correctly, due to various reasons, such as improperly tuned quantitative signal thresholds. If this is the case, we must find the most likely fault that explains the (incorrectly) observed trace, because the observed trace may not be found in any L_f. The standard fault isolation approach (Section 3) assumes the observed trace is always correct, whereas the new robust approach (Section 4) does not make that assumption, in order to handle incorrectly observed fault traces in a robust fashion.

3. EVENT-BASED FAULT ISOLATION

In the standard fault isolation approach, we assume that fault traces are correctly observed.

Assumption 3. All observed fault signatures are correct, i.e., if fault signature σ occurs, it is observed as σ.

Therefore, given Assumptions 1–3, when a fault occurs and we observe a fault trace, this trace must belong to the fault language of at least one fault. The function of the fault isolation algorithm is simply to find which faults are consistent with the observed fault trace.

The fault isolation algorithm is presented as Algorithm 1. Initially, the set of isolated faults, F^*, is set to the complete set of faults, F. The initial observed fault trace λ is the empty event sequence. While new fault signatures are observed, we update the observed fault trace, and reduce F^* to the set of faults consistent with the new trace.

The `FindConsistentFaults` algorithm, presented as Algorithm 2, eliminates from F^* faults that are no longer consistent with the trace extended with σ_i. A fault f is consistent with an observed trace λ if there is a fault trace λ_f in its fault language where λ is a prefix ($\sqsubseteq$), i.e., the fault can generate the observed sequence of events so far. If the fault is indeed consistent, it is retained, otherwise, it is removed from F^*.

Basically, we continue to observe new symbols, and F^* reduces. If the system is *diagnosable*, i.e., all faults are distinguishable from each other (via their fault languages), then F^* will reduce to a single fault. A fault f_i is distinguishable from

Algorithm 2 $F^* \leftarrow$ `FindConsistentFaults`(F^*, λ)

1: **for all** $f \in F^*$ **do**
2: $\quad$ **if** $\neg$ **exist** $\lambda_f \in L_f$ **such that** $\lambda \sqsubseteq \lambda_f$ **then**
3: $\quad\quad F^* \leftarrow F^* - \{f\}$
4: $\quad$ **end if**
5: **end for**

fj in this framework if there is no trace in $\mathcal{L}_{f_i}$ that is a prefix of a trace in $\mathcal{L}_{f_j}$.

Example 1. Consider a set of three faults, $F = \{f_1, f_2, f_3\}$, where $L_{f_1} = \{cab, acb\}$, $L_{f_2} = \{abc, bac\}$, and $L_{f_3} = \{cb, ca, ab\}$. Say that we observe first the fault signature a. Each of the faults may produce a as the first fault signature, so $F^* = \{f_1, f_2, f_3\}$. Say we next observe b. Now, f_1 cannot produce a trace starting with ab, so it is eliminated, and $F^* = \{f_2, f_3\}$. Say we next observe c. Now, f_3 cannot produce a trace beginning with abc, and so f_2 is isolated as the fault.

Let us say we observe a trace that does not belong to any fault language. There are three explanations for this: (*i*) an unknown fault has occurred (violation of Assumption 1), (*ii*) a valid trace is missing from a fault language (violation of Assumption 2), or (*iii*) the trace was observed incorrectly (violation of Assumption 3). For (*i*) and (*ii*), there is nothing that can be done, so we limit ourselves only to situation (*iii*). So, what happens when the trace is observed incorrectly?

Example 2. Consider again the fault set from the previous example. Say we observe c, then we have $F^* = \{f_1, f_3\}$. Say we then observe b, then we have $F^* = \{f_3\}$. Say we then observe a, then we have $F^* = \varnothing$, i.e, all faults were eliminated. One explanation is that the a fault signature was falsely observed (i.e., a false alarm), in which case the true fault is f_3.

The result of an incorrectly observed trace is an incorrect fault isolation result. Either all candidates will be eliminated, as in the example above, or the wrong fault will be isolated (if the observed trace belongs to a fault language of a fault that did not occur). In practice, it is not unlikely that a trace may be incorrectly observed, e.g., from noisy sensor signals, overly sensitive fault detection thresholds, etc. Clearly, Algorithm 1 is not robust in this case. A more robust approach is necessary to handle a violation of Assumption 3.

4. ROBUST EVENT-BASED FAULT ISOLATION

As described in Section 3, Algorithm 1 makes Assumption 3, i.e., there is only one interpretation of an observed trace, which is what was observed. In practice, however, traces may be incorrectly observed, and so we must drop Assumption 3 in order to be robust to this situation, i.e., to make the approach *observation-robust*. In more detail, by observation-robust, we mean that the approach performs optimally when all observations are correct, and its performance degrades gracefully as the number of incorrect observations increases.

In practical terms, this means that the true fault is diagnosed to have the highest probability of being the one that occurred, when all observations are correct. Further, its assigned probability decreases when incorrect observations are encountered, where, up to a certain point, it remains the most probable fault given the observations.

In order to still perform in the face of incorrect observations, we must differentiate between an *observed trace* and an *interpreted trace*. For a given observed trace, there are several potential interpreted traces. An observed trace may or may not belong to any L_f. Any valid interpretation of it, however, must be a prefix of some trace in L_F. That is, given an observed trace, we must generate all correct ways to interpret it, given the set of considered faults. Each interpreted trace will have its own probability and its own diagnosis. Given the set of interpreted traces, their probabilities, and their diagnoses, we can extract a combined diagnosis that provides, for every fault resulting from an interpreted trace, a probability of its occurrence.

Say that so far we have an interpreted trace of λ, and a new symbol σ_i is observed. How do we extend λ given σ_i? We assume there is a known set of signatures, Σ_{σ_i}, that can be observed as σ_i. At a minimum, this set contains σ_i itself. So, when σ_i is observed, it could have been any signature in Σ_{σ_i} that actually occurred. However, only a subset of these can extend λ and be consistent with a given set of faults. To be consistent, they have to be a prefix of some trace found in L_F (since an interpreted trace must belong to L_F).

Example 3. Consider again the set of three faults, $F = \{f_1, f_2, f_3\}$, where $L_{f_1} = \{cab, acb\}$, $L_{f_2} = \{abc, bac\}$, and $L_{f_3} = \{cb, ca, ab\}$. Say that $\Sigma_a = \{a, b\}$, $\Sigma_b = \{b, a\}$, and $\Sigma_c = \{c\}$. Say that the trace bca is observed, what are the possible interpreted traces? First b is observed and that can be interpreted as either a or b; so far the interpreted traces are a and b. Next c is observed, which can be interpreted only as c; so the interpreted traces are ac and bc. Then a is observed, which can be interpreted as either a or b, so the potential interpreted traces are aca, acb, bca, bcb, however, only acb belongs to a fault language and is valid.

Σ_{σ_i} may also contain special signatures that represent false alarms, which we denote using ϵ with a subscript denoting the event associated with the false alarm (e.g., ϵ_a for a false alarm of event a). For example, we could observe some signature σ, but it may be possible that no signature occurred and σ is to be interpreted as a false alarm. In this case, we require a special false alarm signature. The fault languages must include traces that contain false alarm signatures in order for them to be interpreted from an observed trace. Note that such signatures are not required for the standard approach due to Assumption 3. We require also a false alarm "fault" to be included in F, for which its traces contain only false alarm signatures. It is not actually a fault but used to represent the situation where so far, only false alarm signatures have been interpreted from the observed signatures.

Example 4. Consider the same situation as in the previous example, except with false alarm signatures ϵ_a, ϵ_b, and ϵ_c. The fault languages are extended by traces where a, b, and c can be replaced with these signatures, respectively, e.g., L_{f_1}, in addition to cab, has $\epsilon_c ab$, $c\epsilon_a b$, and $ca\epsilon_b$, as well as $\epsilon_a cb$, $\epsilon_b ca$, $\epsilon_a \epsilon_b c$, etc. Here, we have $\Sigma_a = \{a, b, \epsilon_a\}$, $\Sigma_b = \{b, a, \epsilon_b\}$, and $\Sigma_c = \{c, \epsilon_c\}$. We require then also the false alarm fault E, which has all traces of the three signatures ϵ_a, ϵ_b, and ϵ_c. Say again that the trace bca is observed, what are the possible interpreted traces? First b is observed and that can be interpreted as either a, b, or a false alarm in b, ϵ_b. Then c is observed which is really either c or ϵ_c, so the potential interpreted traces are ac, $a\epsilon_c$, $b\epsilon_c$, $\epsilon_b c$, $\epsilon_b \epsilon_c$ (bc is not included since it does not belong to any fault language). Next a is observed which is either a, b, or ϵ_a. The interpreted traces are then acb, $a\epsilon_c b$, $b\epsilon_c a$, $b\epsilon_c \epsilon_a$, $\epsilon_b ca$, $\epsilon_b c\epsilon_a$, $\epsilon_b \epsilon_c a$, and $\epsilon_b \epsilon_c \epsilon_a$.

The algorithm for robust fault isolation is given as Algorithm 3. We keep a set of tuples, $\mathcal{L}$, containing an interpreted trace λ, its probability p, and its diagnosis F^*. Initially, the set contains only one tuple, which is the empty trace ε, with a probability of 1 and the complete fault set F as its diagnosis. When a new signature σ_i is observed (ln. 2), we go through each interpreted trace λ. First, we find all new signatures that would (*i*) belong to Σ_{σ_i}, and (*ii*) can extend λ to produce a valid fault trace (ln. 5). For each of these possible next signatures, we extend the trace with it (ln. 7), assign the new trace's probability (lns. 8–15), and obtain its diagnosis (ln. 16). We then add the new tuple (λ', p', F^*) to the set of new tuples $\mathcal{L}'$ (ln. 17), which replaces $\mathcal{L}$ (ln. 20). Finally, we construct the merged diagnosis $\mathcal{F}^*$, which is a set of tuples of a fault and its probability.

To compute the probability of a trace, we assume that there is a probability of observing the correct signature, p_c. We can compute the probability of the interpreted signature, p_σ, as p_c if it matches the observed signature σ_i. If it does not match, we assume that all other signatures are equally probable, so it is assigned as $(1 - p_c)/(|\Sigma| - 1)$ if σ_i is possible to observe, and $1/|\Sigma|$ if not. The probability of the trace extended by σ is then the probability of the original trace times the probability of σ.

The diagnosis that is merged over all traces is computed as described in Algorithm 4. Each fault is assigned initially a probability of 0. Then, for each interpreted trace, the probability of the fault given that trace, $p(f|\lambda)$, is computed as the a priori probability of the fault divided by the sum of the probabilities of that fault diagnosed for that trace. This probability is then added to the probability of the fault, $p(f)$. After going through all traces, each fault is assigned its total probability. The set $\mathcal{F}^*$ is created by adding tuples for all faults and their probabilities.

Algorithm 3 $\mathcal{F}^* \leftarrow$ `RobustFaultIsolation`(F)

1: $\mathcal{L} \leftarrow \{(\varepsilon, 1, F)\}$
2: **while** σ_i **observed do**
3: $\quad\mathcal{L}' \leftarrow \varnothing$
4: $\quad$**for all** $(\lambda, p, F^*) \in \mathcal{L}$ **do**
5: $\quad\quad\Sigma \leftarrow \{\sigma : \sigma \in \Sigma_{\sigma_i}$ **and exists** $\lambda \in L_{F^*}$ **such that** $\lambda\sigma \sqsubseteq \lambda\}$
6: $\quad\quad$**for all** $\sigma \in \Sigma$ **do**
7: $\quad\quad\quad\lambda' \leftarrow \lambda\sigma$
8: $\quad\quad\quad$**if** $\sigma = \sigma_i$ **then**
9: $\quad\quad\quad\quad p_\sigma \leftarrow p_c$
10: $\quad\quad\quad$**else if** $\sigma_i \in \Sigma$ **then**
11: $\quad\quad\quad\quad p_\sigma \leftarrow (1 - p_c)/(|\Sigma| - 1)$
12: $\quad\quad\quad$**else**
13: $\quad\quad\quad\quad p_\sigma \leftarrow 1/|\Sigma|$
14: $\quad\quad\quad$**end if**
15: $\quad\quad\quad p' \leftarrow p \cdot p_\sigma$
16: $\quad\quad\quad F^* \leftarrow$ `FindConsistentFaults`(F^*, λ')
17: $\quad\quad\quad\mathcal{L}' \leftarrow \mathcal{L}' \cup \{(\lambda', p'F^*)\}$
18: $\quad\quad$**end for**
19: $\quad$**end for**
20: $\quad\mathcal{L} \leftarrow \mathcal{L}'$
21: $\quad\mathcal{L} \leftarrow$ `Prune`$(\mathcal{L})$
22: $\quad\mathcal{F}^* \leftarrow$ `ConstructF`$(F, \mathcal{L})$
23: **end while**

Algorithm 4 $\mathcal{F}^* \leftarrow$ `ConstructF`$(F, \mathcal{L})$

1: $\mathcal{F}^* \leftarrow \varnothing$
2: **for all** $f \in F$ **do**
3: $\quad p(f) \leftarrow 0$
4: **end for**
5: **for all** $(\lambda, p, F^*) \in \mathcal{L}$ **do**
6: $\quad$**for all** $f \in F^*$ **do**
7: $\quad\quad p(f|\lambda) \leftarrow \dfrac{p_f}{\sum_{f' \in F^*} p_{f'}}$
8: $\quad\quad p(f) \leftarrow p(f) + p \cdot p(f|\lambda)$
9: $\quad$**end for**
10: **end for**
11: **for all** $f \in F$ **do**
12: $\quad\mathcal{F}^* \leftarrow \mathcal{F}^* \cup \{(f, p(f))\}$
13: **end for**

Clearly, the number of interpreted traces, in the worst case, grows exponentially with each new observed symbol. Each new symbol can be interpreted in a number of ways and all current interpreted traces need to be extended with all possible interpretations. In order to control the computational complexity of the algorithm, a pruning step is added (ln. 21). Interpreted traces may be removed from $\mathcal{L}$ by, for example, keeping only the N most probable traces, or keeping only traces above a probability threshold p_o. After removing traces from $\mathcal{L}$, the trace probabilities must be normalized.

Example 5. Consider again the scenario in the previous example. The diagnostic tree is shown in Fig. 1. Initially, any of the faults are possible, including the false alarm fault E. The branches in the tree represent the possible interpreted traces from the observed trace bca. The standard approach would have only one branch. We assume that $p_c = 0.9$, and the arrows are labeled with the interpreted symbol and its probability, leading to the new diagnosis and its probability. Since bca does not belong to any fault language, the standard approach would fail, whereas in this approach, we have many potential diagnoses that are ranked probabilistically, depending on the probabilities assigned to the interpreted symbols. For example, take the leftmost branch, where b is correctly observed. This happens with 90% probability, and immediately leads to $\{f_2\}$ as the diagnosis, since no other fault can produce a b as the first signature. Then c is observed. Since there is no fault that can produce bc, the only valid interpretation, given that b was correctly observed, is that c was incorrectly observed and the interpreted signature is ϵ_c, i.e., a false alarm of symbol c. Then a is observed, which can be interpreted only as a or ϵ_a, but not as b since no fault produces two b signatures in any trace. In either case, the diagnosis remains f_2. The rightmost branch, on the other hand, represents the case where all observations were false alarms, and thus the diagnosis is E. For a given fault, its total probability over all interpreted traces can be computed. If we assume that all faults are equally likely, then $p(f_2|bca) = 0.81 + 0.09 + 0.005/3 + 0.0045/3 = 0.9032$.

Clearly, the selection of values for p_c and p_o will determine the final computed probabilities of candidates for a given observed trace. A higher value of p_c will assign a higher probability to the most consistent candidates and a lower value to the remaining candidates, i.e., the candidate probability distribution will have a smaller variance. Similarly, a lower value of p_c will cause the candidate probability distribution to have a larger variance. If p_o is too high, and a trace is incorrectly observed, then it is possible that the correct candidate can be eliminated. Therefore, both p_c and p_o have to be selected to best represent the confidence in the symbol observation process.

5. CASE STUDY

In this section, we describe the application of the new robust event-based fault isolation framework to ADAPT. We use the qualitative event-based fault isolation (QFI) framework developed in (Daigle et al., 2009) and apply the robust methodology to it. We first describe the QFI framework and how it maps into the general event-based framework described earlier, then describe the ADAPT system. Finally, we describe experimental results using data from ADAPT.

5.1. Qualitative Event-Based Fault Isolation

In the QFI framework in (Mosterman & Biswas, 1999; Daigle et al., 2009), signatures capture qualitative deviations in magnitude and slope of residual signals, where a residual is computed as the difference between a measured value of a sensor and its expected (model-predicted) value. So, for a given residual r, we can have six different signatures: (*i*) an increase in magnitude, (*ii*) a decrease in magnitude, (*iii*) an increase in slope, (iv) a decrease in slope, (*v*) a false alarm in the magnitude, and (*vi*) a false alarm in the slope. For each potential fault, we can use a dynamic system model to determine

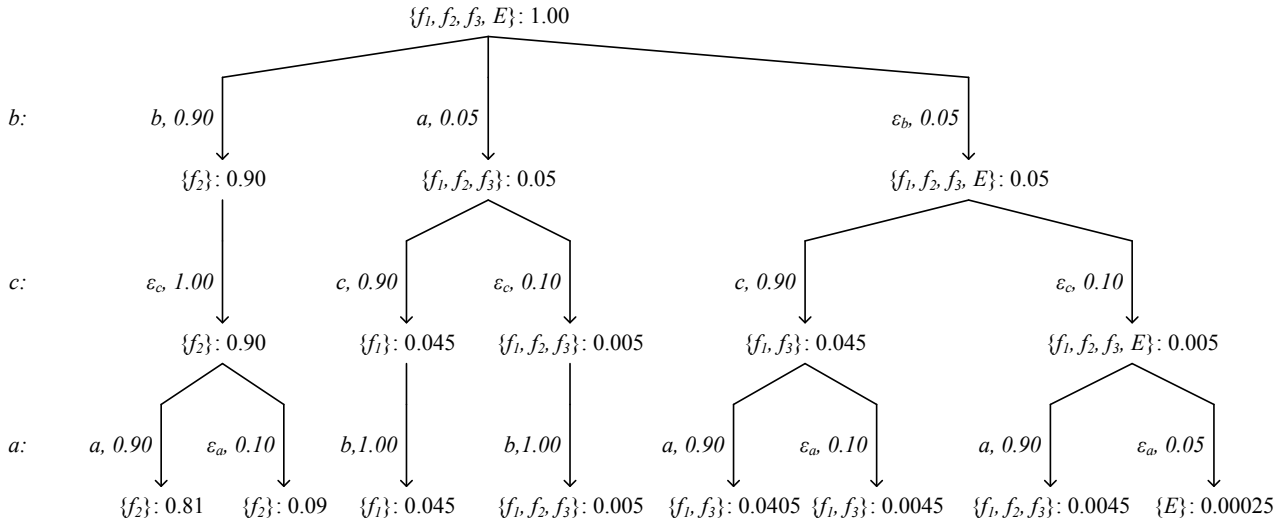

Figure 1. Example diagnostic tree.

which signatures are possible, as described in (Mosterman & Biswas, 1999).

Fault traces in this framework obey a certain set of constraints. First, for a given residual r, the magnitude symbol must always be observed before the slope symbol, and magnitude and slope symbols can be observed only once per residual (including false alarm signatures). Second, the order of signatures between residuals must respect relative residual orderings (Daigle, Koutsoukos, & Biswas, 2007), which express the intuition that faults manifest in some residuals before others. Like signatures, these can be derived from a dynamic system model (Daigle, 2008). Third, once a false alarm signature occurs for the magnitude, we cannot observe any more signatures for that residual. Aside from these restrictions, false alarms can occur at any time. In this framework, fault traces do not need to be precomputed but can be computed online (Daigle et al., 2009).

More information on this framework and its implementation may be found in (Daigle, Roychoudhury, & Bregon, 2013; Daigle, Bregon, & Roychoudhury, 2011). For the purposes of this paper, it suffices to say that we build a dynamic model in order to compute residuals, and these are analyzed in a statistical manner to generate observed signatures. This involves the use of thresholds on the residuals. The major practical problem here is tuning of the thresholds, which can be time-consuming in order to achieve the desired false alarm/missed detection trade-off. If these are not perfectly tuned, signatures can be incorrectly generated. In practice, this is quite difficult, so, using an approach that is robust to incorrect signatures is much desired. We compare two different diagnosers, (*i*) the QED algorithm, which implements the `FaultIsolation` algorithm; and (*ii*) probabilistic QED (pQED), which implements the `RobustFaultIsolation` algorithm. Except for the fault isolation algorithm, the two diagnosers are the same.

5.2. ADAPT

In this paper, we apply our new methodology to the Advanced Diagnostics and Prognostics Testbed (ADAPT), an electrical power distribution system that is representative of those on spacecrafts. ADAPT serves as a testbed through which faults can be injected to evaluate diagnostic algorithms (Poll et al., 2007). ADAPT has been established as a diagnostic benchmark system through the industrial track of the International Diagnostic Competition (DXC) (Kurtoglu et al., 2009; Poll et al., 2011; Sweet et al., 2013). In particular, this paper is focused on diagnosing faults on a subset of ADAPT, called ADAPT-Lite.

A system schematic for ADAPT-Lite is given in Fig. 2. A battery (BAT2) supplies electrical power to several loads, transmitted through several circuit breakers (CB236, CB262, CB266, and CB280) and relays (EY244, EY260, EY281, EY272, and EY275), and an inverter (INV2) that converts dc to ac power. ADAPT-Lite has one dc load (DC485) and two ac loads (AC483 and FAN416). There are sensors throughout the system to report electrical voltage (names beginning with "E"), electrical current ("IT"), and the positions of relays and circuit breakers ("ESH", "ISH"). Finally there is one sensor to report the operating state of a load (fan speed, "ST") and another to report the battery temperature ("TE"). Models and additional details for ADAPT-Lite can be found in (Daigle et al., 2011, 2013).

Our list of potential faults includes failures in the relays, circuit breakers, fan, DC load, and AC load. We consider also

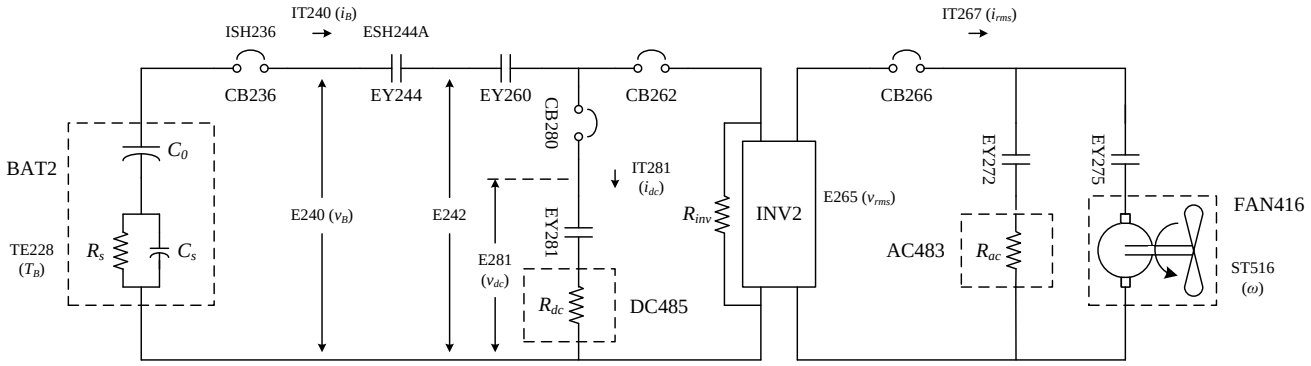

Figure 2. ADAPT-Lite schematic.

under- and over-speed faults of the fan, and offset, drift, and intermittent offset faults in the DC and AC loads.

5.3. Experiments

Using scenarios available from the DXC, we ran QED and pQED on a set of 30 nominal scenarios and 71 fault scenarios. The same fault detectors were used for both algorithms, so that we can show that, when incorrect signatures are generated, pQED performs better than QED, with the same information. The settings are nonoptimal in order to better highlight the differences in the approaches when multiple incorrect observations are encountered; improving the settings would of course improve the performance of both algorithms, but make it harder to compare the performance in nonoptimal conditions.

We first consider an example scenario, to illustrate the different diagnosis approaches. We then summarize the performance of the approaches over all scenarios.

As an example, consider a resistance drift fault in AC483. The fault is injected at 60 s and detected at 63 s with a decrease in IT240. QED reduces the candidate list to a failure in AC483, a positive resistance offset in AC483, a positive resistance drift in AC483, a failure in CB236, CB262, CB266, EY244, and DC485, a resistance increase in DC485, a resistance drift in DC485, a failure in EY244, ER260, EY272, EY275, EY284, FAN416, an under-speed fault in FAN416, and a failure in INV2. A − signature for the slope of the IT240 residual is then computed, for which only the drift faults are consistent. An increase in E242 is detected at 120 s, followed by the generation of a + signature for its slope. QED eliminates all faults, because it expects IT267 to deviate before E242. On the other hand, pQED retains the drift faults as candidates, but lowers their probabilities. Before the E242 deviation, the two drift faults had a probability of 38.77% each. After, the probability reduces to 3.92%, and they are still at the top of the candidate list. With the subsequent signatures for E242, probability decreases, as this is more evidence of other potential faults, but they remain the most probable. However, then E240 deviates, again before IT267 as expected, and this reduces their probability further, and they drop to the eighth and ninth most probable (at this point it is more likely that the detection of a negative slope (rather than no change in slope) was incorrect, and so failures in the circuit breakers and relays become more likely). In this case, no deviation was detected in IT267. With a more sensitive threshold, a deviation in IT267 could have been detected first, and the drift faults would have remained the most probable. Although this is not the most optimal result, at least the true fault was contained in the final diagnosis, albeit not at the highest level of probability.

5.3.1. Summary of Results

Over the nominal scenarios, both algorithms (since they use the same fault detectors) correctly detected a fault (true positives) 69 of 71 times, with 2 missed detections (false negatives). There were no false alarms detected.

For the fault scenarios, QED ends with a list of candidates that are consistent with the observed symbols. Ideally, this list is a singleton, containing the true fault. If, given the available diagnostic information, this is not possible, then we desire that it has the true fault in its final candidate list. In fact, QED never obtains the true fault as the single candidate, as diagnosability is not high enough to achieve that condition.

QED has the correct fault in its candidate list in 24 of 69 scenarios. This means that there are incorrect signatures generated in at least 45 scenarios. This can be improved with better fault detector tuning, however we keep these settings in order to demonstrate the improvement pQED provides. In 32 of these 45 scenarios, QED actually eliminates all faults, as no faults were consistent with the (incorrect) observations.

For pQED, we used $p_c = 90\%$, and pruned candidates with probability less than 0.1%. If pQED does not prune, then it will always have the correct candidate in its candidate list (but perhaps with a low probability assignment). With the pruning threshold used, pQED has the correct candidate in

its final list 63 of 69 times, which is a significant improvement over QED. For the 6 times in which it did not have the true fault, there were too many incorrect observations, bringing down the probability of the true fault low enough that all traces containing the fault were pruned.

Of course, it is not enough the pQED has the correct fault in its list, as this depends solely on the pruning threshold. We are interested in the probability assignment of the true fault within the final candidate list. pQED diagnoses the true fault as the fault with highest probability 38 of 69 times. This is better than the 24 of 69 times for QED. Since QED does not rank its final candidates, pQED's result is actually significantly better and more useful. For the times when the true fault is not ranked the highest, it is at least contained in the final candidate list for most of the time.

6. CONCLUSIONS

In this paper, we presented a robust approach to event-based fault isolation that drops the observation correctness assumption in order to improve robustness of fault isolation when events are incorrectly observed. We applied this framework to a qualitative event-based fault isolation framework. Experiments using real data from an electrical power system testbed demonstrated the approach and its improved robustness.

Future work will focus on extending the approach to multiple fault isolation, and extending the probability framework to account for conditional probabilities.

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.

REFERENCES

Alonso-Gonzalez, C., Moya, N., & Biswas, G. (2011). Dynamic bayesian network factors from possible conflicts for continuous system diagnosis. In *Proc. of the 14th int. conf. on advances in ai* (pp. 223–232). Berlin: Springer-Verlag.

Cordier, M.-O., & Dousson, C. (2000, June). Alarm driven monitoring based on chronicles. In *Proceedings of the 4th symposium on fault detection supervision and safety for technical processes* (p. 286-17291).

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

Daigle, M., Bregon, A., & Roychoudhury, I. (2011, October). Qualitative Event-based Diagnosis with Possible Conflicts: Case Study on the Third International Diagnostic Competition. In *Proceedings of the 22nd international workshop on principles of diagnosis* (p. 285-292). Murnau, Germany.

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

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

Daigle, M., Roychoudhury, I., & Bregon, A. (2013, October). Qualitative event-based diagnosis with possible conflicts: Case study on the fourth international diagnostic competition. In *Proceeedings of the 24th international workshop on principles of diagnosis* (p. 230-235).

Hofbaur, M., & Williams, B. (2002, May). Hybrid diagnosis with unknown behavioral modes. In *Proceedings of the 13th international workshop on principles of diagnosis* (pp. 97–105).

Koscielny, J., & Zakroczymski, K. (2000). Fault isolation method based on time sequences of symptom appearance. In *Proceedings of ifac safaprocess.* Budapest, Hungary.

Kurtoglu, T., Narasimhan, S., Poll, S., Garcia, D., Kuhn, L., de Kleer, J., ... Feldman, A. (2009, June). First international diagnosis competition – DXC'09. In *Proceedings of 20th international workshop on principles of diagnosis* (p. 383-396).

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., & Brownston, L. (2007, May). HyDE — a general framework for stochastic and hybrid model-based diagnosis. In *Proc. of the 18th int. workshop on principles of diagnosis* (pp. 162–169).

Pernestål, A. (2009). *Probabalistic fault diagnosis with automotive applications*. Unpublished doctoral dissertation, Linköping University.

Poll, S., de Kleer, J., Abreau, R., Daigle, M., Feldman, A., Garcia, D., ... Sweet, A. (2011, October). Third international diagnostics competition – DXC'11. In *Proc. of the 22nd international workshop on principles of diagnosis* (pp. 267–278).

Poll, S., Patterson-Hine, A., Camisa, J., Nishikawa, D., Spirkovska, L., Garcia, D., ... Lutz, R. (2007, May). Evaluation, selection, and application of model-based diagnosis tools and approaches. In *AIAA infotech@aerospace 2007 conference and exhibit.*

Puig, V., Quevedo, J., Escobet, T., & Pulido, B. (2005). On the Integration of Fault Detection and Isolation in Model Based Fault Diagnosis. In *Proceedings of the 16th international workshop on principles of diagnosis, dx05* (p. 227-232). Pacific Grove, CA, USA.

Ricks, B., & Mengshoel, O. (2009, September). Methods for

probabilistic fault diagnosis: an electrical power system case study. In *Annual conference of the prognostics and health management society (phm09).* San Diego, USA.

Roychoudhury, I. (2009). *Distributed diagnosis of continuous systems: Global diagnosis through local analysis*. Unpublished doctoral dissertation, Vanderbilt University.

Roychoudhury, I., Biswas, G., & Koutsoukos, X. (2010). Distributed diagnosis in uncertain environments using dynamic bayesian networks. In *18th mediterranean conference on control & automation (med),* (pp. 1531–1536).

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

Sweet, A., Feldman, A., Narasimhan, S., Daigle, M., & Poll, S. (2013, September). Fourth international diagnostic competition – DXC'13. In *Proc. of the 24th international workshop on principles of diagnosis* (pp. 224–229).

Ying, J., Kirubarajan, T., Pattipati, K., & Patterson-Hine, A. (2000, Nov). A hidden Markov model-based algorithm for fault diagnosis with partial and imperfect tests. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, *30*(4), 463-473. doi: 10.1109/5326.897073

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.