System Diagnosability Analysis using Model-Based Diagnosis Tools

Gregory Provan

Rockwell Science Center, 1049 Camino dos Rios, Thousand Oaks, CA 91320
805-373-4726
gmprovan@rsc.rockwell.com

ABSTRACT

Computing the diagnosability of a discrete-valued system (such as an avionics system), or conversely, a set of test vectors to efficiently determine system diagnosability, is a well-known task within the area of system diagnostics. There are a number of approaches that have been adopted for this task, and many tools have been developed and are available commercially. This article describes a new approach for this task, using techniques developed within the model-based diagnostics (MBD) community. The benefits of this new approach are: (1) the same model used for system design and analysis can be used for diagnosability testing; and (2) a diagnosability model (or set of test vectors) can be compiled from the MBD model, without having to have a model for design and one for diagnosability.

Keywords: Model-based diagnosis, Diagnosability, Testability.

1. INTRODUCTION

Given the increasing complexity of diagnostic tasks that need to be performed on 21-st Century Aerospace Applications, greater demands are being place on modeling techniques and diagnostics algorithms. One key area where good algorithms are necessary is that of diagnosability (or testability), which aims to (1) identify the faults that are diagnosable given a particular suite of embedded sensors, and (2) generate a set of tests to efficiently isolate faults. This area has spawned a number of model-based testability solutions, such as [8, 9, 12]

The field of AI also has addressed similar concerns, leading to the development of what is known as Model-Based Diagnostics, or MBD [1, 2, 5, 14]. Model-based diagnostics provide the capability for high-fidelity, component-based diagnostics that isolate multiple-faults with great precision. However, model-based diagnostics representations and algorithms are typically both memory- and computation-intensive. This approach has focused primarily on just high-fidelity fault isolation, and has ignored the issues of diagnosability/testability. This paper presents some of our latest results in using efficient model-based diagnostics representations and algorithms for diagnosability.

This article describes the theoretical underpinnings of our representations and algorithms, as applied to diagnosability analysis. Our model-based diagnostic system employs the causal network approach, which is a well-known model-based diagnostic approach. We extend this approach to create a reasoning system that provides the advantages of model-based reasoning and scales to the requirements of modern aerospace applications.

We show in this article how the MBD approach is a strict extension to the approach developed within the testability community. Hence, in comparison to the testability approach, the MBD approach can be applied to more general, i.e. more complex, models, and can provide strong guarantees of diagnostic soundness and completeness, given the model; however, this comes at the price of greater model-building and computational demands. Note that when the testability and MBD models are essentially the same, the MBD approach has complexity similar to that of the testability approach, but can provide stronger guarantees about the quality of the diagnostics it can generate than can the testability approach. In contrast, testability approaches cannot provide such soundness and completeness guarantees of diagnostic isolation. From this perspective, the MBD approach is strictly better than the testability approach, although MBD is typically computationally more expensive than testability approaches. In contrast, the MBD community has typically focused on diagnostics generation, and does not have as sophisticated a set of test-sequence generation procedures as that found within the testability community, e.g. [9, 12]. In this article we focus solely on diagnostic isolation, i.e., in generating the fault matrix (or fault dictionary) [12].

The article is organized as follows. Section 2 reviews the notation used in standard diagnosability tools. Section 3 presents an overview of our modeling representation, causal networks. Section 4 described how we use MBD to generate a fault matrix, and Section 5 summarizes our results.

2. DIAGNOSABILTY ANALYSIS

2.1 Representation

This section introduces the notation used in [8, 12] for diagnosability (testability) analysis.

There are two primary variables, a set T of *n* binary tests, and a set F of *m* failure sources. The model M consists of a binary causal graph describing the failure space of how failures propagate to monitoring points. The model consists of nodes $V=T\cup F$, and edges $E\subseteq T\times F$. The goal of standard testability is to compute a binary test matrix B of dimension $n\times m$, with tests as rows and failure sources (which we call single-fault diagnoses) in the columns. B can be computed using reachability analysis algorithms using the model M [12].

Note that an MBD model can be restricted to the testability model, in which case the models can generate identical results.

2.2 Example

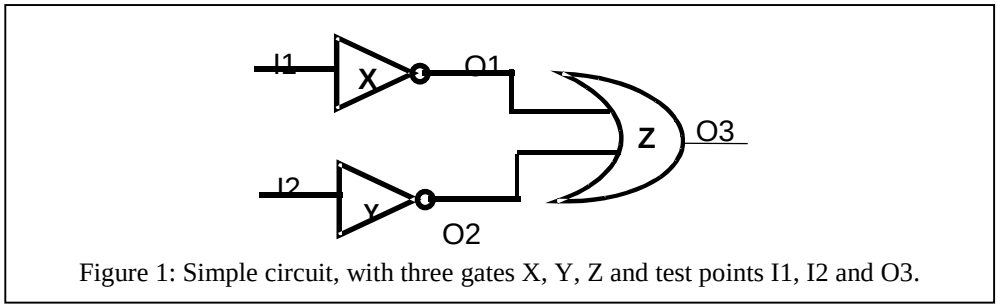

Figure 1: Simple circuit, with three gates X, Y, Z and test points I1, I2 and O3.

This section outlines an example, which we will use throughout the article. This example consists of a simple electrical circuit, with two inverters and an OR-gate, as shown in Figure 1. In this example, we assume that we want to diagnose faults in the three gates (X, Y, Z), using $\{t, f\}$ observations that can be taken of the inputs I1 and I2, and the output O3. We assume that the outputs for the inverters (O1 and O2) are not observable. We refer to Ξ as the set of unobservable variables in any model. For simplicity, we assume that each gate is either functioning normally (OK), or is faulty, in which case the gate outputs the opposite of what it should output under normal operation.

The testability model that is constructed for this example is shown in Figure 2.

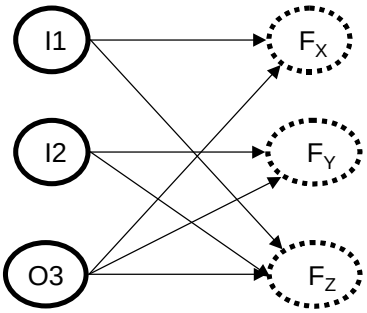

Figure 2: Testability model for simple circuit, a bipartite graph.

3. MODEL-BASED DIAGNOSIS REPRESENTATION

This section introduces our MBD representation, causal networks [2], and compares and contrasts it with testability notation. Note that, given a theoretical definition of MBD, e.g. [5], there are two main implementations of the technology, those based on the Assumption-Based TMS [15], such as GDE [16], and those based on causal networks [2].

3.1 Comparison of Testability and MBD

MBD is a generalization of Testability in several important ways.

1. The set T of binary tests is generalized to a set of multi-valued tests.
2. The set F of binary failure sources is generalized to a set of multi-valued failure modes.
3. The testability model, which consists of a binary causal graph, is generalized to a Directed-Acyclic graph (DAG), which encodes not just the space of failures, but also the space of normal modes. Because both normal and failure modes are modeled in a fairly detailed manner, the MBD model can be used for both discrete simulation and for testing the results of fault injection (simulating the effects of failures). This is why this class of model can be used for system design as well.

Prior to defining our notation, we show how a causal network model is used to encode the simple circuit example, in order to provide important intuitions about this representation. A causal network model encodes the causality underlying a system. We encode such causal dependence of node V1 on node V2 in a graphical model using a directed arc from node V2 to node V1. In our circuit example, each gate is causally dependent on its input(s) and on the operating mode of the gate, which in our case is OK or faulty. Hence the output O1 of inverter X is dependent on the input I1 and on the failure mode variable F_X.

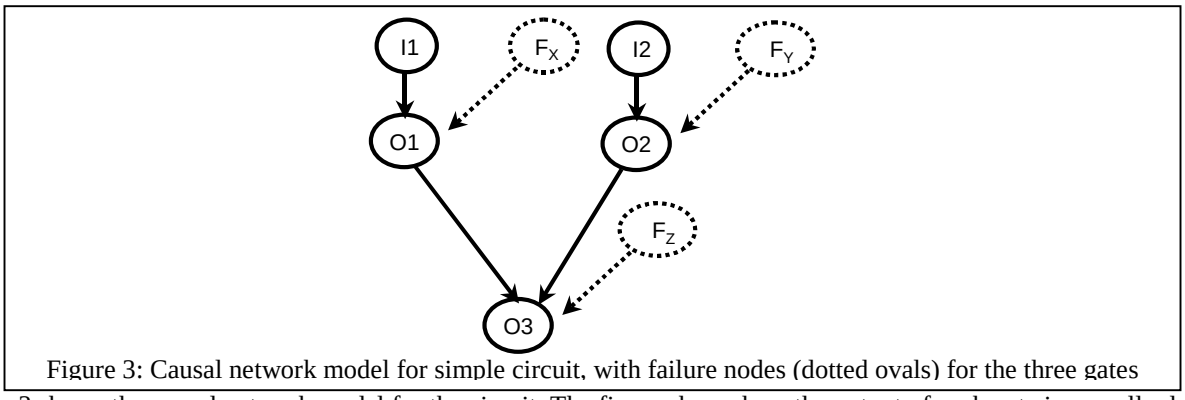

Figure 3: Causal network model for simple circuit, with failure nodes (dotted ovals) for the three gates

Figure 3 shows the causal network model for the circuit. The figure shows how the output of each gate is causally dependent on its inputs and on its failure mode.

There are several differences between the graphical models specified for testability models and causal networks. First, the edge orientation is different: in a causal network a node for a failure mode is only a parent node, whereas it is always a child node in a testability graph. A causal network encodes the "forward" causality from faults and inputs to outputs (which is why is can be used for simulation), whereas a testability model encodes "reverse" causality, like that of a rule-based model. A testability model cannot perform simulation, since all causality is directed from observations to faults, and this causality is not invertible.

Second, a testability model is typically restricted to binary form (i.e., only observable to fault links), whereas a causal network does not have this restriction. If this restriction is imposed, then the causal network will still encode "forward" causality (i.e., fault to observable links), as shown in Figure 4. When encoding probabilities, this type of model is called naïve-Bayes, and for our purposes we call it a naïve diagnostic model.

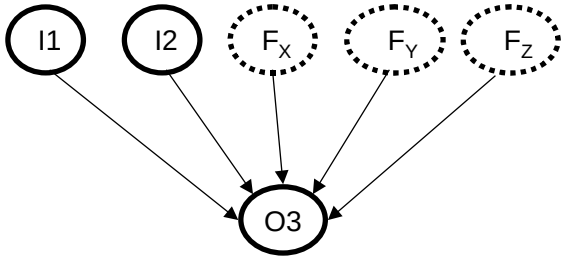

Figure 4: Restricted causal network model for simple circuit, with only observable and fault nodes.

Third, an MBD approach typically makes use of a test that provides data for all measurable variables, e.g., for all sensors. This is because the MBD algorithms are focused on generating all sound diagnoses given a single test. This contrasts with the testability approach, in which tests often consist of a single measurement point (or sensor), and the notion is that there will be a sequence of tests, at the end of which the diagnoses can be isolated. There are extensions to the MBD approach [4, 7] in which a sequence of additional tests are used, in case the initial test could not isolate the diagnosis sufficiently well, e.g., if the size of the ambiguity set was too large.

3.2 Representation

We now outline a diagnostic model in terms of a model-based representation, that of causal networks [2]. As much as possible, we adopt the terminology of the testability approach, to provide a means for comparing MBD with testability.

We define a *model specification M* for a diagnostic model using the tuple ($\mathbf{\Sigma}$,**F**,G,Ψ):

Definition 1: A *model specification* is defined using a tuple ($\mathbf{\Sigma}$,**F**,G,Ψ), where $\mathbf{\Sigma}\cup$**F** is the set of system variables, G is a directed acyclic graph consisting of nodes corresponding to $\mathbf{\Sigma}\cup$**F**, and Ψ is a quantification of the variables, i.e., a set of multi-valued propositional sentences constructed from $\mathbf{\Sigma}\cup$**F**.

Definition 2 [System Variables]: We partition the system variables into two sets: $\mathbf{\Sigma}$ is a set of atomic propositions, **F** is a set of distinguished atomic propositions called assumables. The atomic propositions $\mathbf{\Sigma}$ describe the entities in the model, which include system components. The assumables F are the variables describing the operating characteristics of the components that we want to diagnose.

We further partition the atomic propositions $\mathbf{\Sigma}$ into a set ***S*** of measurable variables (sensors and actuators), and a set Ξ of unmeasurable variables: $\mathbf{\Sigma} = \Xi\cup\boldsymbol{S}$. The measurable variables are the variables that represent sensors and actuators, i.e. components whose values we can measure (sensor) or components whose values we can set (actuator).

The graph G specifies the causal relationships among the variables, and consists of a directed acyclic graph over the variables. Φ(X) denotes the parents of any node X in the directed acyclic graph G. The graphical model consists of nodes V=T$\cup$F$\cup\Xi$, and edges E$\subseteq$ (T$\cup$F$\cup\Xi$)$\times$(T$\cup\Xi$).

The assumables are the discrete-valued variables describing the modes, or operating characteristics, of the components. The elements of **F** can have finite domains, e.g., {OK, failed-high, failed-low}. In the graph G, each assumable is associated with a single node *N* in G, which is called the corresponding component. We view the assumable as an equation-selector, i.e., it selects the set of equations specifying the behavior of *N* based on the mode of *N*. Note that if we only know about normal modes, then we can define only equations for normal modes. In this case, an anomaly is any deviation outside of the set of normal modes. Defining fault modes provides the ability to perform more accurate simulation.

If each element of **F** consists only of binary-valued variables, i.e., OK and broken (or $\neg$OK), then we say that **F** is mode-less. If the elements of **F** can have finite domains, e.g., {OK,failed-high,failed-low}, then we say that **F** has modes, and we are performing diagnosis with modes. In this case each system failure-mode can exhibit multiple normal and multiple fault modes.

Definition 3 [Observation]: A diagnostic system observation T is an instantiation of the values of (a subset of) the discrete-valued measurables of the diagnostic model.[1]

Definition 4 [Quantification]: Ψ is a set of propositional sentences constructed from atoms in $\mathbf{\Sigma}$ and **F**. More specifically, we associate with each variable X in $\mathbf{\Sigma}$ a subset Ψ_X of sentences in Ψ, such that X is the consequent of each sentence in Ψ_X, and Φ(X) constitute the antecedents of each sentence in Ψ_X. The antecedents of any variable are defined using a directed acyclic graph G.

The equations define a set of constraints over the possible values of each variable. We can specify a model in a number of ways, such as using logic [5], constraints [14], or causal networks [2]. Here we define the system model using causal networks, which specify a graph G and quantification consisting of a set of multi-valued propositional clauses. Note that testability models typically do not contain such a set of propositional statements describing a model.

3.3 Diagnostic Inference

Model-based inference is as follows: given SD$\cup$T, find the failure-mode specification A* such SD$\cup$T$\cup$A* does not entail $\perp$. Here A* is an assignment of mode values to all system failure-modes (or more generally behavioral modes).

[1] This notion of observation corresponds to a test in testability terminology.

Computing the value of A* is an NP-hard task, since it searches over all subsets of mode-instantiations. Since straightforward search of the space of diagnoses can be of exponential size, we apply a focusing algorithm to conduct a best-first search over the space of diagnoses [2]. This focusing algorithm assigns a set Σ of costs to the fault modes, and then uses an algebra to perform the best-first search over the diagnosis space. Σ is derived from the failure history, e.g., component reliability data. Following [2], we can characterize the algebra using the triple $(\Sigma,\oplus,\geq_{\oplus})$, where $\oplus$ is a cost addition operation and $\geq_{\oplus}$ is a cost total ordering satisfying:

- $\oplus$ is commutative, associative and has a zero element;
- $j \geq_{\oplus} i$ iff $i \oplus k = j$ for some k.

An example of this is $\langle \mathbf{Z},+,\geq \rangle$.

The full specification for the diagnosis task requires the tuple $\langle \Psi$, T, Dx-Alg, $(\Sigma,\oplus,\geq_{\oplus})\rangle$, where

- Ψ is the system model,
- T is the input observation,
- Dx-Alg is the diagnosis inference algorithm, and
- $(\Sigma,\oplus,\geq_{\oplus})$ characterizes the focusing algorithm.

The inference system CNETS [3] is an implementation of the causal network algorithms, and has been applied to many areas, such as avionics [13], rocket engines [10], and fault-tolerant systems [11].

MBD approaches provide a number of strong guarantees. For example, given the model, the causal network MBD approach guarantees soundness and completeness of the computed diagnoses [2].

4. USING MODEL-BASED DIAGNOSIS FOR DIAGNOSABILITY ANALYSIS

4.1 Procedure

Our diagnosability procedure consists of three main steps:

1. For each fault F in a set of faults to test, **F**, simulate the associated test- (or sensor-vector) T. This produces a set of pairs of faults and test-vectors, {<F, T>}.
2. For each test-vector, compute the diagnosis α associated with that vector, generating the tuple {<F, T, α>}.
3. Compute the diagnosability statistics given the tuples.

To facilitate this procedure, we have two models: a simulation model M_S, and a diagnostic model M_D. Each model is optimized for its particular task, either simulation or diagnosis, as it is possible to have a single model to perform both tasks.

We compute a number of diagnosability statistics. Our key concern is to specify the difference between the real fault F and the computed fault (as represented by the diagnosis α); we average this value over a large suite of real faults **F**. If α=F, then we have perfect diagnosability. In many cases there is an ambiguity group (set of disjunctive diagnoses) computed for α, so we must compute the degree of diagnosability that we have. If $\alpha=\varnothing$, then that fault is undiagnosable.

In this document we focus on single faults, although our approach can compute all multiple-fault diagnoses possible in a system [2, 5].

4.2 Example

We now show how we use our MBD approach to compute the diagnosability of the simple circuit. We first use our simulation model to generate test-vectors for all faults and for all input settings for I1 and I2. In other words, for each of the 4 possible settings for I1 and I2, we simulate a fault in gates X, Y and Z, and record the value of the output O3. Table 1 summarizes the results of this simulation. For example, with I1 and I2 both set to *t*, simulating a fault F_X will create a setting of *t* for O3, as shown in the third row of Table 1. Note that there are 4 additional setting of I1, I2 and O3 that correspond to normal conditions, and we do not represent them in Table 1.

		F_X	F_Y	F_Z
I1	**I2**	O3	O3	O3
t	***t***	*t*	*t*	*t*
t	***f***	*t*	*f*	*f*
f	***t***	*f*	*t*	*f*
f	***f***	*t*	*t*	*f*

Table 1: Summary of results of fault simulation for simple circuit.

We now transform this table into a table with a set of tests, where each test is an assignment of values to (I1, I2, O3), together with the diagnosis corresponding to each test. Table 2 shows this table, with the first column showing the diagnosis, and the last three columns the settings for each test.

Diagnosis	Test	I1	I2	O3
$F_X \vee F_Y \vee F_Z$	**T1**	t	t	t
$F_Y \vee F_Z$	**T2**	t	f	f
$F_X \vee F_Z$	**T3**	f	t	f
F_Z	**T4**	f	f	f

Table 2: Table depicting relationship between diagnoses and tests for simple circuit.

At this point, we can further rearrange this data to create a fault matrix. This fault matrix is shown in Table 3. In this matrix a 1 denotes that the test detects the fault shown in the first column.

	T1	***T2***	***T3***	***T4***
F_X	1		1	
F_Y	1	1		
F_Z	1	1	1	1

Table 3: Fault matrix for simple circuit.

In this example, note that for simplicity of exposition we have restricted our analysis to single-faults. Multiple-fault cases would yield more detailed tables, and we do not describe such cases here.

4.3 Diagnosability Measures

A broad range of diagnosability measures can be computed for a system, and many of these are described in [12]. In this document, we focus on a few measures. We assume that a diagnosis α consists of an ambiguity group with k fault-mode elements, i.e., $\alpha = \bigvee_i F_i$, $i=1,\ldots k$. The measures that we can compute are:

Ambiguity group size:
We define the size of an ambiguity group to be $|\alpha_k|$. We call a singleton an ambiguity group of size 1.

Average (mean) ambiguity group size:
If we have m_i ambiguity groups of size $i=|\alpha_i|$, then the mean group size is given by:

$$\overline{\alpha} = \frac{\sum_{i=1}^{n} m_i \mid \alpha_i \mid}{\sum_{i=1}^{n} m_i}$$

Note that the more accurate the fault isolation, the closer this measure will be to 1.

Probability of Fault Isolation: Isolate(S, T_k)

We have a system S with *n* components that can fail, a component *i* has ω_i fault modes, and a given test T_k. We define an indicator variable c_{ij} that measures whether, given a simulation of F(i,j), fault mode j for component i, F(i,j) is included in the ambiguity group α returned by the diagnostic algorithm, i.e. $F(i,j) \cap \alpha \neq \varnothing$.

$c_{ij} = 1$ if $F(i,j) \cap \alpha \neq \varnothing$
0 otherwise.

$$Isolate(S,T_k) = \frac{\sum_i \sum_j c_{ij}}{n \sum_i \omega_i}$$

Probability of *Unique* Fault Isolation: UIsolate(S, T_k)

We have a system S with *n* components that can fail, and component *i* has ω_i fault modes. We define an indicator variable c_{ij} that measures whether, given a simulation of F(i,j), fault mode j for component i, F(i,j) is equal to the singleton ambiguity group α^* returned by the diagnostic algorithm, i.e. $F(i,j) = \alpha^*$.

$u_{ij} = 1$ if $F(i,j) = \alpha^*$
0 otherwise.

$$UIsolate(S,T_k) = \frac{\sum_i \sum_j u_{ij}}{n \sum_i \omega_i}$$

Unique Fault Coverage:

This is the proportion of faults that can be isolated uniquely, of all those that are isolatable:

$$UCoverage(S,T_k) = \frac{\sum_i \sum_j u_{ij}}{\sum_i \sum_j c_{ij}}$$

(Relative) Efficiency of Test Setting

By default we can choose some Test setting T* to be the standard, and identify efficiency of other settings with respect to this gold standard.

$$Efficiency(T_j,T^*) = \frac{Isolate(S,T_j)}{Isolate(S,T^*)}$$

We now compute a few statistics for the circuit example. Table 4 shows the four possible diagnoses given tests T1 to T4. From this data it is simple to compute the average ambiguity group size to be 4. The probability of fault isolation is 1 for each test. Table 4 also shows that the different tests have different probabilities of unique fault isolation.

Diagnosis	Test	Prob(Unique Isolation)
$F_X \vee F_Y \vee F_Z$	**T1**	0.33
$F_Y \vee F_Z$	**T2**	0.5
$F_X \vee F_Z$	**T3**	0.5
F_Z	**T4**	1

Table 4: Description of diagnoses and corresponding tests.

In addition we can compute overall diagnosability, averaged over all tests. Table 5 shows the probability of isolation to a given number of components, a statistic related to the average ambiguity group size. This table shows that given tests T1 to T4, it is not possible to uniquely isolate faults in components X and Y; to do so, we would need to be able to test the values of O2 and O3, as given in Figure 1.

	1	*2*	*≥3*
$\mathbf{F_X}$	0	1	0
$\mathbf{F_Y}$	0	1	0
$\mathbf{F_Z}$	1	0	0

Table 5: Overall diagnosability, averaged over all tests.

4.3 Complexity Issues

This section briefly outlines issues concerned with complexity of using MBD for computing diagnoses and test matrices. In general, the diagnosis task is NP-hard, whether one uses an MBD approach [6] or the digraph model approach used in testability analysis [9]. The diagnosability approach breaks the diagnosis task up into two steps: (1) generating a binary test matrix, and (2) designing a testing strategy that unambiguously isolates the failure sources with minimum expected test cost. In contrast, the MBD approach aims to perform both steps simultaneously, with the assumption that all test data is available at one time. Because of this difference in tasks being performed, it is somewhat difficult to directly compare the two approaches. The most relevant comparison is that of comparing equal tasks.

For example, if we restrict our model to what we referred to as a naïve diagnostic model (see Figure 4), a model similar to the testability model, the complexity of computing single-fault or multiple-fault diagnoses (given a fixed set of tests) is comparable to the complexity of using the digraph model approach. Note that finding the optimal test sequence for this task is NP-complete [9]. Comparing other like diagnostic tasks yields comparable complexity results for both MBD and diagnosability approaches. The complexity of using MBD is thus comparable to that of using standard testability approaches, when the same type of underlying model is used. Note, however, that MBD is typically used for more complex models [5, 14].

5. SUMMARY AND CONCLUSIONS

We have described an approach that uses Model-Based Diagnosis representations and algorithms to compute diagnosability results, i.e., a fault matrix. We have shown the equivalence of these results to those derivable from standard testability approaches, using a simple example for illustrative purposes. We have also shown how the MBD approach is a strict extension of the testability approach, but reverts to the testability approach when restricted to the class of models typically used in testability analysis.

There are a number of tradeoffs involved in choosing an approach for diagnosability. The theory of both diagnosability and MBD are mature, and there are several commercial products available based on both approaches. The main tradeoff concerns model fidelity and model-building/computational cost. If the only modeling information available or necessary consists of the binary fault/monitoring-point relationships, then the diagnosability approach is probably preferred. However, such models cannot be used for system design and simulation, and are limited in the diagnostic guarantees that can be made. Given more detailed modeling information, the MBD approach can encode additional information, providing higher-fidelity diagnostics in correspondance with the additional model data. MBD models can be used for system design and simulation, and can provide guarantees about the soundness and completeness of the diagnostics, given the model. The tradeoff is the extra effort to construct the higher-fidelity models, and the computational cost of generating the higher-fidelity diagnostics. If simulation models are being generated during the design process, then the MBD data can be folded into the design process and diagnostics can be computed with relatively little extra cost.

6. REFERENCES

[1] L. Console, W. Hamscher, and J. deKleer, Eds. *Readings in Model-Based Diagnosis*, Morgan-Kaufmann Publishers, 1992.

[2] A. Darwiche, Model-based diagnosis using structured system descriptions. *J. of AI Research*, **8**:165- 222, June 1998.

[3] Darwiche, "Model-Based Diagnosis Under Real-World Constraints, " *AI Magazine*, Summer 2000.

[4] J. deKleer and O. Raiman, “Trading off the Costs of Inference vs. Probing in Diagnosis”, ”, Principles of Diagnosis

Workshop, New Paltz, NY, pp. 81-86, October 1994.
[5] O. Dressler and P. Struss, "The Consistency-based Approach to Automated Diagnosis of Devices", in *Principles of Knowledge Representation*, G. Brewka (editor), CSLI Publications, 267-311, 1996.
[6] T. Eiter and G. Gottlob, 1992. On the Complexity of Propositional Knowledge Base Revision, Updates and Counterfactuals, *Artificial Intelligence*, **57**: 227-270.
[7] D. Heckerman, J. Breese and K. Rommelse, "Troubleshooting under Uncertainty", Principles of Diagnosis Workshop, New Paltz, NY, pp. 121-131, October 1994.
[8] K. Pattipatti, V. Raghavan, M. Shakeri, S. Deb and R. Shrestha, "TEAMS: Testability Engineering and Maintenance System", *Proc. American Control Conference*, Baltimore, MD, June 1994.
[9] K. Pattipatti and M. Alexanderidis, "Application of Heuristic Search and Information Theory to Sequential Fault Diagnosis", *IEEE Trans. On Systems, Man and Cyb*., **20**(4): 872-887, July 1990.
[10] G. Provan and D. Glover. "A Model-based Approach to Model Integration and Diagnosis," *J. AI Communications* **12**: 19-32, 1999.
[11] G. Provan and Y-L. Chen, "Model-Based Fault Tolerant Control Reconfiguration for Discrete-Event Systems", *Proc. CACSD*, Anchorage, AK, 2000.
[12] W. R. Simpson and J. W. Sheppard, *System Test and Diagnosis*, Kluwer Academic Publishers, 1994.
[13] C. Sitter and G. Provan, 1998. "Advanced Maintenance using Causal Networks", Proc. DASC-98, Seattle, WA.
[14] P. Struss, "What's in SD? Towards a Theory of Modeling for Diagnosis", in L. Console, Hamscher, W. and J. deKleer, Eds. *Readings in Model-Based Diagnosis*, Morgan-Kaufmann Publishers, pp. 419-449, 1992.
[15] J. de Kleer, "An Assumption-based Truth Maintenance System", *Artificial Intelligence*, **28**: 127-161, 1986.
[16] J. de Kleer and B. Williams, "Diagnosing Multiple Faults", *Artificial Intelligence*, **32**: 97-130, 1987.