

DIAGNOSING MULTIPLE FAULTS¹

Johan de Kleer and Brian C. Williams
Xerox Palo Alto Research Center
3333 Coyote Hill Road, Palo Alto CA 94304 USA

Abstract

Diagnostic tasks require determining the differences between a model of an artifact and the artifact itself. The differences between the manifested behavior of the artifact and the predicted behavior of the model guide the search for the differences between the artifact and its model. The diagnostic procedure presented in this paper is model-based, inferring the behavior of the composite device from knowledge of the structure and function of the individual components comprising the device. The system (GDE — General Diagnostic Engine) has been implemented and tested on many examples in the domain of troubleshooting digital circuits.

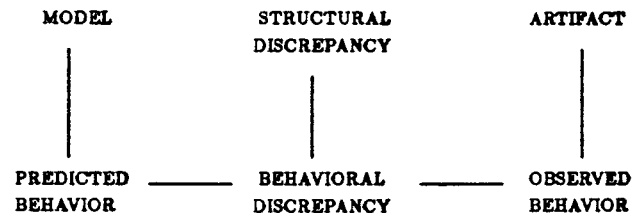
This research makes several novel contributions: First, the system diagnoses failures due to multiple faults. Second, failure candidates are represented and manipulated in terms of minimal sets of violated assumptions, resulting in an efficient diagnostic procedure. Third, the diagnostic procedure is incremental, exploiting the iterative nature of diagnosis. Fourth, a clear separation is drawn between diagnosis and behavior prediction, resulting in a domain (and inference procedure) independent diagnostic procedure. Fifth, GDE combines model-based prediction with sequential diagnosis to propose measurements to localize the faults. The normally required conditional probabilities are computed from the structure of the device and models of its components. This capability results from a novel way of incorporating probabilities and information theory into the context mechanism provided by Assumption-Based Truth Maintenance.

1 Introduction

Engineers and scientists constantly strive to understand the differences between physical systems and their models. Engineers troubleshoot mechanical systems or electrical circuits to find broken parts. Scientists successively refine a model based on empirical data during the process of theory formation. Many everyday commonsense reasoning tasks involve finding the difference between models and reality.

¹ This paper is a correction (as of March 22, 1992) of one which first appeared in the journal, *Artificial Intelligence* 32(1987) 97-130.

Figure 1: Model-Artifact difference.



Diagnostic reasoning requires a means of assigning credit or blame to parts of the model based on observed behavioral discrepancies. If the task is troubleshooting, then the model is presumed to be correct and all model-artifact differences indicate part malfunctions. If the task is theory formation, then the artifact is presumed to be correct and all model-artifact differences indicate required changes in the model (Figure 1).

Usually the evidence does not admit a unique model-artifact difference. Thus, the diagnostic task requires two phases. The first, mentioned above, identifies the set of possible model-artifact differences. The second proposes evidence-gathering tests to refine the set of possible model-artifact differences until they accurately reflect the actual differences.

This view of diagnosis is very general, encompassing troubleshooting mechanical devices and analog and digital circuits, debugging programs, and modeling physical or biological systems. Our approach to diagnosis is also independent of the inference strategy employed to derive predictions from observations.

Earlier work (see Section 6) on model-based diagnosis concentrated on determining a single faulty component that explains all the symptoms. This paper extends that research by diagnosing systems with multiple failed components, and by proposing a sequence of measurements which efficiently localize the failing components.

When one entertains the possibility of multiple faults, the space of potential candidates grows exponentially with the number of faults under consideration. This work is aimed specifically at developing an efficient general method, referred to as the General Diagnostic Engine (GDE), for diagnosing failures due to any number of simultaneous faults. To achieve the needed efficiency, GDE exploits the features of Assumption-Based Truth Maintenance (ATMS) [8]. This is the topic of the first

half of the paper.

Usually, additional measurements are necessary to isolate the set of components which are actually faulted. The best next measurement is the one which will, on average, lead to the discovery of the faulted set of components in a minimum number of measurements. Unlike other probabilistic techniques which require a vast number of conditional probabilities, GDE need only be provided with the a priori probabilities of individual component failure. Using an ATMS, this probabilistic information is incorporated into GDE such that it is straightforward to compute the conditional probabilities of the candidates, as well as the probabilities of the possible outcomes of measurements, based on the faulty device's model. This combination of probabilistic inference and Assumption-Based Truth Maintenance enables GDE to apply a minimum entropy method [1] to determine what measurement to make next: the best measurement is the one which minimizes the expected entropy of candidate probabilities resulting from the measurement. Somewhat surprisingly, the expected entropy of a measurement can be computed directly from the current probabilities of the differing outcomes of the measurement. This is the topic of the second half of the paper.

1.1 Troubleshooting circuits

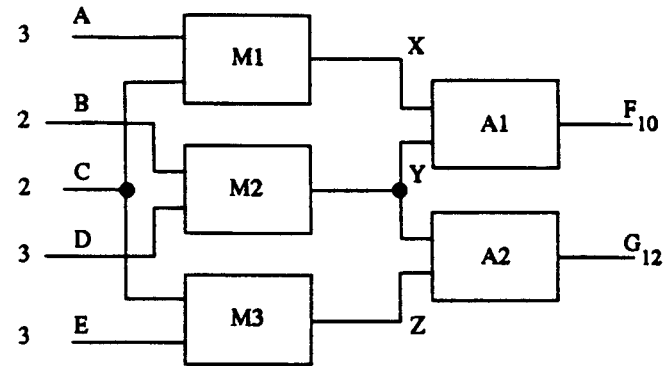
For troubleshooting circuits, the diagnostic task is to determine why a correctly designed piece of equipment is not functioning as it was intended; the explanation for the faulty behavior being that the particular piece of equipment under consideration is at variance in some way with its design (e.g., a set of components is not working correctly or a set of connections is broken). To troubleshoot a system, a sequence of measurements must be proposed, executed and then analyzed to localize this point of variance, or fault. The task for the diagnostician is to use the results of measurements to identify the cause of the variance when possible, and otherwise to determine which additional measurements must be taken.

For example, consider the circuit in Figure 2, consisting of three multipliers, M_1 , M_2 , and M_3 , and two adders, A_1 and A_2 . The inputs are $A = 3$, $B = 2$, $C = 2$, $D = 3$, and $E = 3$, and the outputs are measured showing that $F = 10$ and $G = 12$.² From these measurements it is possible to deduce that at least one of the following sets of components is faulty (each set is referred to as a candidate and is designated by [...]): $[A_1]$, $[M_1]$, $[A_2, M_2]$, or $[M_2, M_3]$. Furthermore, measuring X is likely to produce the most useful information in further isolating the faults. Intuitively, X is optimal because it is the only measurement that can differentiate between the two highly probable singleton candidates: $[A_1]$ and $[M_1]$.

Next the value of X is measured and the result is used to reduce the size of the candidate set. The candidate

²This circuit is also used by both [6] and [15] in explaining their systems.

Figure 2: A familiar circuit.



generation-measurement process continues until a single high probability candidate remains.

1.2 Some basic presuppositions

Although GDE considers multiple faults and probabilistic information, it shares many of the basic presuppositions of other model-based research. We presume that the act of taking a measurement (i.e., making an observation) has no affect on the faulty device. We presume that once a quantity is measured to be a certain value, that the quantity remains at that value. This is equivalent to assuming that no component's (correct or faulty) functioning depends on the passage of time. For example, this rules out flip-flops as well as intermittent components which spontaneously change their behavior. We presume that if a component is faulty, the distribution of input-output values becomes random (i.e., contains no information). We do not presume that if a component is faulty, that it must be exhibiting this faulty behavior — it may exhibit faulty behavior on some other set of inputs. These presuppositions suggest future directions for research, and we are extending GDE in these directions.

2 A Theory of Diagnosis

The remainder of this paper presents a general, domain-independent, diagnostic engine (GDE) which, when coupled with a predictive inference component provides a powerful diagnostic procedure for dealing with multiple faults. In addition the approach is demonstrated in the domain of digital electronics, using propagation as the predictive inference engine.

2.1 Model-artifact differences

The model of the artifact describes the physical structure of the device in terms of its constituents. Each type of constituent obeys certain behavioral rules. For example, a simple electrical circuit consists of wires, resistors and so forth, where wires obey Kirchoff's Current Law, resistors obey Ohm's Law, and so on. In diagnosis, it

is given that the behavior of the artifact differs from its model. It is then the task of the diagnostician to determine what these differences are.

The model for the artifact is a description of its physical structure, plus models for each of its constituents. A constituent is a very general concept, including components, processes and even steps in a logical inference. In addition, each constituent has associated with it a set of one or more possible model-artifact differences which establishes the grain size of the diagnosis.

Diagnosis takes (1) the physical structure, (2) models for each constituent, (3) a set of possible model-artifact differences and (4) a set of measurements, and produces a set of candidates, each of which is a set of differences which explains the observations.

Our diagnostic approach is based on characterizing model-artifact differences as assumption violations. A constituent is guaranteed to behave according to its model only if none of its associated differences are manifested, i.e., all the constituent's assumptions hold. If any of these assumptions are false, then the artifact deviates from its model, thus, the model may no longer apply. An important ramification of this approach ([4; 6; 10; 15; 32]) is that we need only specify correct models for constituents — explicit fault models are not needed.

Reasoning about model-artifact differences in terms of assumption violations is very general. For example, in electronics an assumption might be the correct functioning of each component and the absence of any short circuits; in a scientific domain a faulty hypothesis; in a commonsense domain an assumption such as persistence, defaults or Occam's Razor.

2.2 Detection of symptoms

We presume (as is usually the case) that the model-artifact differences are not directly observable.³ Instead, all assumption violations must be inferred indirectly from behavioral observations. In Section 2.7 we present a general inference architecture for this purpose, but for the moment we presume an inference procedure which makes behavioral predictions from observations and assumptions without being concerned about the procedure's details.

Intuitively, a *symptom* is any difference between a prediction made by the inference procedure and an observation. Consider our example circuit. Given the inputs, $A = 3$, $B = 2$, $C = 2$, $D = 3$, and $E = 3$, by simple calculation (i.e., the inference procedure), $F = X + Y = A \times C + B \times D = 12$. However, F is measured to be 10. Thus "F is observed to be 10, not 12" is a symptom. More generally, a symptom is any inconsistency detected by the inference procedure, and may occur between two predictions (inferred from distinct measurements) as well as a measurement and a prediction (inferred from some other measurements).

³In practice the diagnostician can sometimes directly observe a malfunctioning component by looking for a crack or burn mark.

2.3 Conflicts

The diagnostic procedure is guided by the symptoms. Each symptom tells us about one or more assumptions that are possibly violated (e.g., components that may be faulty). Intuitively, a *conflict* is a set of assumptions which support a symptom, and thus leads to an inconsistency. In this electronics example, a conflict is a set of components which cannot all be functioning correctly. Consider the example symptom "F is observed to be 10, not 12." The prediction that $F = 12$ depends on the correct operation of A_1 , M_1 , and M_2 , i.e., if A_1 , M_1 , and M_2 were correctly functioning, then $F = 12$. Since F is not 12, at least one of A_1 , M_1 , and M_2 is faulted. Thus the set $\langle A_1, M_1, M_2 \rangle$ is a conflict for the symptom (conflicts are indicated by $\langle \dots \rangle$). Because the inference is monotonic with the set of assumptions, the set $\langle A_1, A_2, M_1, M_2 \rangle$, and any other superset of $\langle A_1, M_1, M_2 \rangle$ are conflicts as well; however, no subsets of $\langle A_1, M_1, M_2 \rangle$ are necessarily conflicts since all the components in the conflict were needed to predict the value at F .

A measurement might agree with one prediction and yet disagree with another, resulting in a symptom. For example, starting with the inputs $B = 2$, $C = 2$, $D = 3$, and $E = 3$, and assuming A_2 , M_2 , and M_3 are correctly functioning we calculate G to be 12. However, starting with the observation $F = 10$, the inputs $A = 3$, $C = 2$, and $E = 3$, and assuming that A_1 , A_2 , M_1 , and M_3 , (i.e., ignoring M_2) are correctly functioning we calculate $G = 10$. Thus, when G is measured to be 12, even though it agrees with the first prediction, it still produces a conflict based on the second: $\langle A_1, A_2, M_1, M_3 \rangle$.

For complex domains any single symptom can give rise to a large set of conflicts, including the set of all components in the circuit. To reduce the combinatorics of diagnosis it is essential that the set of conflicts be represented and manipulated concisely. If a set of components is a conflict, then every superset of that set must also be a conflict. Thus the set of conflicts can be represented concisely by only identifying the minimal conflicts, where a conflict is minimal if it has no proper subset which is also a conflict. This observation is central to the performance of our diagnostic procedure. The goal of *conflict recognition* is to identify the complete set of minimal conflicts.⁴

2.4 Candidates

A *candidate* is a particular hypothesis for how the actual artifact differs from the model. For example "A2 and M2 are broken" is a candidate for the two symptoms observed for our example circuit. Ultimately, the goal of diagnosis is to identify, and refine, the set of candidates consistent with the observations thus far.

A candidate is represented by a set of assumptions (indicated by $[\dots]$). The assumptions explicitly mentioned

⁴Representing the conflict space in terms of minimal conflicts is analogous to the idea of version spaces for representing plausible hypotheses in single concept learning [22].

are false, while the ones not mentioned are true. A candidate which explains the current set of symptoms is a set of assumptions such that if every assumption fails to hold, then every known symptom is explained. Thus each set representing a candidate must have a non-empty intersection with every conflict.

For electronics, a candidate is a set of failed components, where any components not mentioned are guaranteed to be working. Before any measurements have been taken we know nothing about the circuit. The candidate space is the set of candidates consistent with the observations. The size of the initial candidate space grows exponentially with the number of components. Any component could be working or faulty, thus the candidate space for Figure 2 initially consists of $2^5 = 32$ candidates.

It is essential that candidates be represented concisely as well. Notice that, like conflicts, candidates have the property that any superset of a possible candidate for a set of symptoms must be a possible candidate as well. Thus the candidate space can be represented by the *minimal candidates*. Representing and manipulating the candidate space in terms of minimal candidates is crucial to our diagnostic approach. Although the candidate space grows exponentially with the number of potentially faulted components, it is usually the case that the symptoms can be explained by relatively few minimal candidates.

The goal of *candidate generation* is to identify the complete set of minimal candidates. The space of candidates can be visualized in terms of a subset-superset lattice (Figure 3). The minimal candidates then define a boundary such that everything from the boundary up is a valid candidate, while everything below is not.

Given no measurements every component might be working correctly, thus the single minimal candidate is the empty set, \square , which is the root of the lattice at the bottom of Figure 3.

To summarize, the set of candidates is constructed in two stages: conflict recognition and candidate generation. Conflict recognition uses the observations made along with a model of the device to construct a complete set of minimal conflicts. Next, candidate generation uses the set of minimal conflicts to construct a complete set of minimal candidates. Candidate generation is the topic of the next section, while conflict recognition is discussed in Section 2.6.

2.5 Candidate generation

Diagnosis is an incremental process; as the diagnostician takes measurements he continually refines the candidate space and then uses this to guide further measurements. Within a single diagnostic session the total set of candidates must decrease monotonically. This corresponds to having the minimal candidates move monotonically up through the candidate superset lattice towards the candidate containing all components. Similarly, the total set of conflicts must increase monotonically. This corresponds to having the minimal conflicts move monotonically

ically down through a conflict superset lattice towards the conflict represented by the empty set. Candidates are generated incrementally, using the new minimal conflict(s) and the old minimal candidate(s) to generate the new minimal candidate(s).

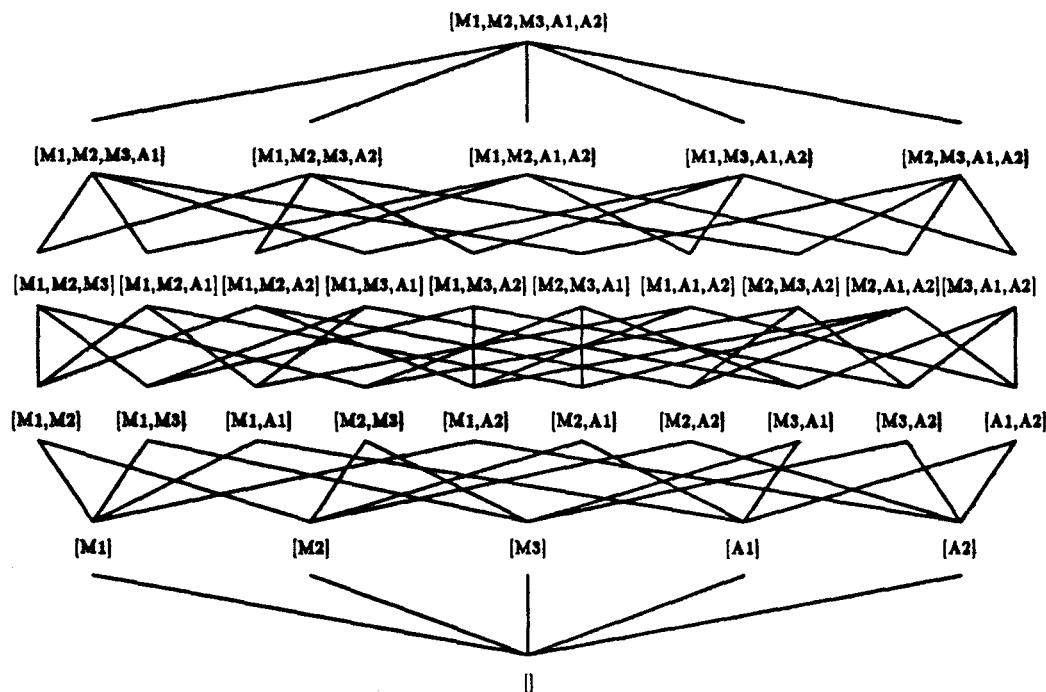
The set of minimal candidates is incrementally modified as follows. Whenever a new conflict is discovered, any previous minimal candidate which does not explain the new conflict is replaced by one or more superset candidates which are minimal based on this new information. This is accomplished by replacing the old minimal candidate with a set of tentative new minimal candidates each of which contains the old candidate plus one assumption from the new conflict. Any tentative new candidate which is subsumed or duplicated by another is eliminated; the remaining candidates are added to the set of new minimal candidates.

Consider our example. Initially there are no conflicts, thus the minimal candidate \square (i.e., everything working) explains all observations. We have already seen that the single symptom " $F = 10$ not 12 " produces one conflict $\langle A_1, M_1, M_2 \rangle$. This rules out the single minimal candidate \square . Thus, its immediate supersets containing one assumption of the conflict $[A_1]$, $[M_1]$ and $[M_2]$ are considered. None of these are duplicated or subsumed as there were no other old minimal candidates. The new minimal candidates are $[A_1]$, $[M_1]$, and $[M_2]$. This situation is depicted with the lattice in Figure 4. All candidates above the line labeled by the conflict " $C1: \langle A_1, M_1, M_2 \rangle$ " are valid candidates.

The second conflict (inferred from observation $G = 12$), $\langle A_1, A_2, M_1, M_3 \rangle$, only eliminates minimal candidate $[M_2]$; the unaffected minimal candidates $[M_1]$, and $[A_1]$ remain. However, to complete the set of minimal candidates we must consider the immediate supersets of $[M_2]$ which cover the new conflict: $[A_1, M_2]$, $[A_2, M_2]$, $[M_1, M_2]$, and $[M_2, M_3]$. Each of these candidates explains the new conflict, however, $[A_1, M_2]$ and $[M_1, M_2]$ are supersets of the minimal candidates $[A_1]$ and $[M_1]$, respectively. Thus the new minimal candidates are $[A_2, M_2]$, and $[M_2, M_3]$, resulting in the minimal candidate set: $[A_1]$, $[M_1]$, $[A_2, M_2]$, and $[M_2, M_3]$. The line labeled by conflict " $C2: \langle A_1, A_2, M_1, M_3 \rangle$ " in Figure 4 shows the candidates eliminated by the observation $G = 12$ alone, and the line labeled " $C1 \& C2$ " shows the candidates eliminated as a result of both measurements ($F = 10$ and $G = 12$). The minimal candidate which split the lattice into valid and eliminated candidates are circled.

Candidate generation has several interesting properties. First, the set of minimal candidates may increase or decrease in size as a result of a measurement; however, a candidate, once eliminated can never reappear. As measurements accumulate, eliminated minimal candidates are replaced by larger candidates. Second, if an assumption appears in every minimal candidate (and thus every candidate), then that assumption is necessarily false (i.e., that component is necessarily faulted). Third, the presupposition that there is only a single fault

Figure 3: Initial candidate space for the circuit example.



(exploited in all previous model-based troubleshooting strategies), is equivalent to assuming all candidates are singletons. In this case, the set of candidates can be obtained by intersecting all the conflicts.

2.6 Conflict recognition strategy

The remaining task involves incrementally constructing the conflicts used by candidate generation. In this section we first present a simple model of conflict recognition. This approach is then refined into a more efficient strategy.

A conflict can be identified by selecting a set of assumptions, referred to as an *environment*, and testing whether or not they are inconsistent with the observations.⁵ If they are, then the inconsistent environment is a conflict. This requires an inference strategy $C(OBS, ENV)$ which, given the set of observations OBS made thus far and the environment ENV, determines whether the combination is consistent. In our example,

⁵An environment should not be confused with a candidate or conflict. An environment is a set of assumptions all of which are assumed to be true (e.g., the environment $\{M_1, M_2\}$ indicates that M_1 and M_2 are assumed to be working correctly), a candidate is a set of assumptions all of which are assumed to be false (e.g., the candidate $[M_1, M_2]$ indicates that M_1 and M_2 are not functioning correctly). A conflict is a set of assumptions, at least one of which is false (e.g., the conflict (M_1, M_2) indicates that M_1 or M_2 is faulted). Intuitively an environment is the set of assumptions that define a "context" in a deductive inference engine, in this case the engine is used for prediction and the assumptions are about the lack of particular model-artifact differences.

after measuring $F = 10$, and before measuring $G = 12$, $C(\{F = 10\}, \{A_1, M_1, M_2\})$ (leaving off the inputs) is false indicating the conflict $\langle A_1, M_1, M_2 \rangle$. This approach is refined as follows:

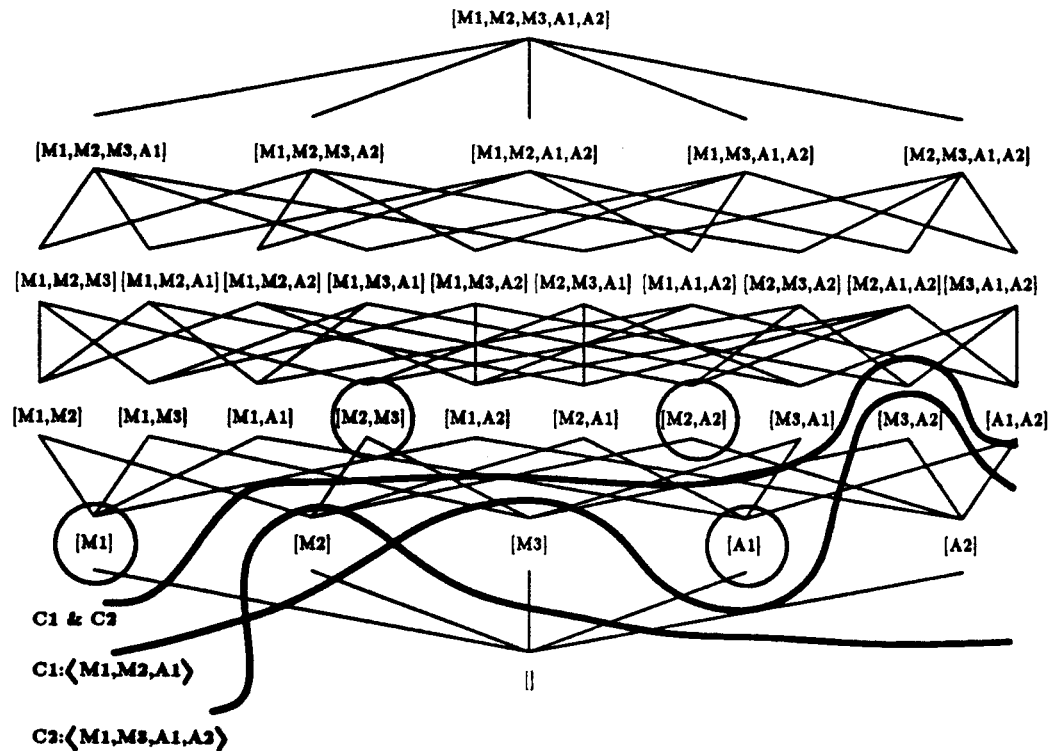
Refinement 1: Exploiting minimality. To identify the set of minimal inconsistent environments (and thus the minimal conflicts), we begin our search at the empty environment, moving up along its parents. This is similar to the search pattern used during candidate generation. At each environment we apply $C(OBS, ENV)$ to determine whether or not ENV is a conflict. Before a new environment is explored, all other environments which are a subset of the new environment must be explored first. If the environment is inconsistent then it is a minimal conflict and its supersets are not explored. If an environment has already been explored or is a superset of a conflict then C is not run on the environment and its supersets are not explored.

We presume the inference strategy operates entirely by inferring hypothetical predictions (e.g., values for variables in environments given the observations made). Let $P(OBS, ENV)$ be all behavioral predictions which follow from the observations OBS given the assumptions ENV. For example, $P(\{A = 3, B = 2, C = 2, D = 3\}, \{A_1, M_1, M_2\})$ produces $\{A = 3, B = 2, C = 2, D = 3, X = 6, Y = 6, F = 12\}$.

C can now be implemented in terms of P . If P computes two distinct values for a quantity (or more simply both x and $\neg x$), then a symptom is manifested and ENV is a conflict.

Refinement 2: Monotonicity of measurements. If in-

Figure 4: Candidate space after measurements.



put values are kept constant, measurements are cumulative and our knowledge of the circuit's structure grows monotonically. Given a new measurement M , $P(\text{OBSU}\{M\}, \text{ENV})$ is always a superset of $P(\text{OBS}, \text{ENV})$. Thus if we cache the values of every P , when a new measurement is made we need only infer the incremental addition to the set of predictions.

Refinement 3: Monotonicity for assumptions. Analogous to refinement 2, the set of predictions grows monotonically with the environment. If a set of predictions follows from an environment, then the addition of any assumption to that environment only expands this set. Therefore $P(\text{OBS}, \text{ENV})$ contains $P(\text{OBS}, E)$ for every subset E of ENV . This makes the computation of $P(\text{OBS}, \text{ENV})$ very simple if all its subsets have already been analyzed.

Refinement 4: Redundant inferences. P must be run on a large number of (overlapping) environments. Thus the same rule will be executed over and over again on the same facts. All of this overlap can be avoided by utilizing ideas of Truth Maintenance such that every inference is recorded as a dependency and no inference is ever performed twice [14].

Refinement 5: Exploiting the sparseness of the search space. The four refinements allow the strategy to ignore (i.e., to the extent of not even generating its name) any environment which doesn't contain some interesting inferences absent in every one of its subsets. If every

environment contained a new unique inference, then we would still be faced computationally with an exponential in the number of potential model-artifact differences. However, in practice, as the components are weakly connected, the inference rules are weakly connected. Therefore, it is more efficient to associate environments with rules than vice versa. Our strategy depends on this empirical property. For example, in electronics the only assumption sets of interest are sets of components which are connected and whose signals interact — typically circuits are explicitly designed so that component interactions are limited.

2.7 Inference procedure architecture

To completely exploit the ideas discussed in the preceding section we need to modify and augment the implementation of P . We presume that P meets (or can be modified to) the two basic criteria for utilizing truth maintenance: (1) A dependency (i.e., justification) can be constructed for each inference, and (2) belief or disbelief in a datum is completely determined by these dependencies. In addition, we presume that, during processing, whenever more than one inference is simultaneously permissible, that the actual order in which these inferences are performed is irrelevant and that this order can be externally controlled (i.e., by our architecture). Finally, we presume that the inference procedure is monotonic. Most AI inference procedures meet these

four general criteria. For example, many expert rule-based systems, constraint propagation, demon invocation, taxonomic reasoning, qualitative simulations, natural deduction systems, and many forms of resolution theorem-proving fit this general framework.

We associate with every prediction, V , the set of environments, $ENV(S(V))$, from which it follows (i.e., $ENV(S(V)) \equiv \{env | V \in P(OBS, env)\}$). We call this set the *supporting environments of the prediction*. Exploiting the monotonicity property, it is only necessary to represent the minimal (under subset) supporting environments.

Consider our example after the measurements $F = 10$ and $G = 12$. In this case we can calculate $X = 6$ in two different ways. First, $Y = B \times D = 6$ assuming M_2 is functioning correctly. Thus, one of its supporting environments is $\{M_2\}$. Second, $Y = G - Z = G - (C \times E) = 6$ assuming A_2 and M_3 are working. Therefore the supporting environments of $Y = 6$ are $\{\{M_2\}, \{A_2, M_3\}\}$. Any set of assumptions used to derive $Y = 6$ is a superset of one of these two.

By exploiting dependencies no inference is ever done twice. If the supporting environments of a prediction change, then the supporting environments of its consequents are updated automatically by tracing the dependencies created when the rule was first run. This achieves the consequences of a deduction without rerunning the rule.

We control the inference process such that whenever more than one rule is runnable, the one producing a prediction in the smaller supporting environment is performed first. A simple agenda mechanism suffices for this. Whenever a symptom is recognized, the environment is marked a conflict and all rule execution stops on that environment. Using this control scheme predictions are always deduced in their minimal environment, achieving the desired property that only minimal conflicts (i.e., inconsistent environments) are generated.

In this architecture P can be incomplete (in practice it usually is). The only consequence of incompleteness is that fewer conflicts will be detected and thus fewer candidates will be eliminated than the ideal — no candidate will be mistakenly eliminated.

3 Circuit Diagnosis

Thus far we have described a very general diagnostic strategy for handling multiple faults, whose application to a specific domain depends only on the selection of the function P . In this section, we demonstrate the power of this approach, by applying it to the problem of circuit diagnosis.

For our example we make a number of simplifying pre-suppositions. First, we assume that the model of a circuit is described in terms of a circuit topology plus a behavioral description of each of its components. Second, that the only type of model-artifact difference considered is whether or not a particular component is working correctly. Finally, all observations are made in terms of

measurements at a component's terminals.

Measurements are expensive, thus not every value at every terminal is known. Instead, some values must be inferred from other values and the component models. Intuitively, symptoms are recognized by propagating out locally through components from the measurement points, using the component models to deduce new values. The application of each model is based on the assumption that its corresponding component is working correctly. If two values are deduced for the same quantity in different ways, then a coincidence has occurred. If the two values differ then the coincidence is a symptom. The conflict then consists of every component propagated through from the measurement points to the point of coincidence (i.e., the symptom implies that at least one of the components used to deduce the two values is inconsistent). Note however, if the two coinciding values are the same, then it is not necessarily the case that the components involved in the predictions are functioning correctly. Instead, it may be that the symptom simply does not manifest itself at that point. Also, it might be that one of these components is faulty, but does not manifest its fault, given the current set of inputs. (For example, an inverter with an output stuck at one will not manifest a symptom given an input of zero.) Thus if the coinciding values are in agreement then no information is gained.

3.1 Constraint propagation

Constraint propagation [36; 37] operates on cells, values, and constraints. Cells represent state variables such as voltages, logic levels, or fluid flows. A constraint stipulates a condition that the cells must satisfy. For example, Ohm's Law, $v = iR$, is represented as a constraint among the three cells v , i , and R . Given a set of initial values, constraint propagation assigns each cell a value that satisfies the constraints. The basic inference step is to find a constraint that allows it to determine a value for a previously unknown cell. For example, if it has discovered values $v = 2$ and $i = 1$, then it uses the constraint $v = iR$ to calculate the value $R = 2$. In addition, the propagator records R 's dependency on v , i and the constraint $v = iR$. The newly recorded value may cause other constraints to trigger and more values to be deduced. Thus, constraints may be viewed as a set of conduits along which values can be propagated out locally from the inputs to other cells in the system. The recorded dependencies trace out a particular path through the constraints that the inputs have taken. A symptom is manifested when two different values are deduced for the same cell (i.e., a logical inconsistency is identified). In this event dependencies are used to construct the conflict.

Sometimes the constraint propagation process terminates leaving some constraints unused and some cells unassigned. This usually arises as a consequence of insufficient information about device inputs. However, this can also arise as the consequence of logical incompleteness in the propagator.

In the circuit domain, the behavior of each component is modeled as a set of constraints. For example, in analyzing analog circuits the cells represent circuit voltages and currents, the values are numbers, and the constraints are mathematical equations. In digital circuits, the cells represent logic levels, the values are 0 and 1, and the constraints are boolean equations.

Consider the constraint model for the circuit of Figure 2. There are ten cells: $A, B, C, D, E, X, Y, Z, F$, and G , five of which are provided the observed values: $A = 3, B = 2, C = 2, D = 3$ and $E = 3$. There are three multipliers and two adders each of which is modeled by a single constraint: $M_1 : X = A \times C, M_2 : Y = B \times D, M_3 : Z = C \times E, A_1 : F = X + Y$, and $A_2 : G = Y + Z$. The following is a list of deductions and dependencies that the constraint propagator generates (a dependency is indicated by (component : antecedents)):

$$\begin{aligned} X &= 6 (M_1 : A = 3, C = 2) \\ Y &= 6 (M_2 : B = 2, D = 3) \\ Z &= 6 (M_3 : C = 2, E = 3) \\ F &= 12 (A_1 : X = 6, Y = 6) \\ G &= 12 (A_2 : Y = 6, Z = 6) \end{aligned}$$

A symptom is indicated when two values are determined for the same cell (e.g., measuring F to be 10 not 12). Each symptom leads to new conflict(s) (e.g., in this example the symptom indicates a conflict $\langle A_1, M_1, M_2 \rangle$).

This approach has some important properties. First, it is not necessary for the starting points of these paths to be inputs or outputs of the circuit. A path may begin at any point in the circuit where a measurement has been taken. Second, it is not necessary to make any assumptions about the direction that signals flow through components. In most digital circuits a signal can only flow from inputs to outputs. For example, a subtractor cannot be constructed by simply reversing an input and the output of an adder since it violates the directionality of signal flow. However, the directionality of a component's signal flow is irrelevant to our diagnostic technique; a component places a constraint between the values of its terminals which can be used in any way desired. To detect discrepancies, information can flow along a path through a component in any direction. For example, although the subtractor does not function in reverse, when we observe its outputs we can infer what its inputs must have been.

3.2 Generalized constraint propagation

Each step of constraint propagation takes a set of antecedent values and computes a consequent. We have built a constraint propagator within our inference architecture which explores minimal environments first. This guides each step during propagation in an efficient manner to incrementally construct minimal conflicts and candidates for multiple faults.

Consider our example. We ensure that propagations in subset environments are performed first, thereby

guaranteeing that the resulting supporting environments and conflicts are minimal. We use $[x, e_1, e_2, \dots]$ to represent the assertion x with its associated supporting environments. Before any measurements or propagations take place, given only the inputs, the data base consists of: $[A = 3, \{\}], [B = 2, \{\}], [C = 2, \{\}], [D = 3, \{\}],$ and $[E = 3, \{\}]$. Observe that when propagating values through a component, the assumption for that component is added to the dependency, and thus to the supporting environment(s) of the propagated value. Propagating A and C through M_1 we obtain: $[X = 6, \{M_1\}]$. The remaining propagations produce: $[Y = 6, \{M_2\}], [Z = 6, \{M_3\}], [F = 12, \{A_1, M_1, M_2\}],$ and $[G = 12, \{A_2, M_2, M_3\}]$.

Suppose we measure F to be 10. This adds $[F = 10, \{\}]$ to the data base. Analysis proceeds as follows (starting with the smaller environments first): $[X = 4, \{A_1, M_2\}]$, and $[Y = 4, \{A_1, M_1\}]$. Now the symptom between $[F = 10, \{\}]$ and $[F = 12, \{A_1, M_1, M_2\}]$ is recognized indicating a new minimal conflict: $\langle A_1, M_1, M_2 \rangle$. Thus the inference architecture prevents further propagation in the environment $\{A_1, M_1, M_2\}$ and its supersets. The propagation goes one more step: $[G = 10, \{A_1, A_2, M_1, M_3\}]$. There are no more inferences to be made.

Next, suppose we measure G to be 12. Propagation gives: $[Z = 6, \{A_2, M_2\}], [Y = 6, \{A_2, M_3\}], [Z = 8, \{A_1, A_2, M_1\}],$ and $[X = 4, \{A_1, A_2, M_3\}]$. The symptom " $G = 12$ not 10" produces the conflict $\langle A_1, A_2, M_1, M_3 \rangle$. The final data-base state is shown below.⁶

$$\begin{aligned} &[F = 12, \{A_1, M_1, M_2\}] \\ &[A = 3, \{\}] \\ &[B = 2, \{\}] \\ &[C = 2, \{\}] \\ &[D = 3, \{\}] \\ &[E = 3, \{\}] \\ &[F = 10, \{\}] \\ &[G = 12, \{\}] \\ &[X = 4, \{A_1, M_2\} \{A_1, A_2, M_3\}] \\ &[\quad 6, \{M_1\}] \\ &[Y = 4, \{A_1, M_1\}] \\ &[\quad 6, \{M_2\} \{A_2, M_3\}] \\ &[Z = 8, \{A_1, A_2, M_1\}] \\ &[\quad 6, \{M_3\} \{A_2, M_2\}] \end{aligned}$$

This results in two minimal conflicts:

$$\begin{aligned} &\langle A_1, M_1, M_2 \rangle \\ &\langle A_1, A_2, M_1, M_3 \rangle \end{aligned}$$

The algorithm discussed in Section 2.5 uses the two minimal conflicts to incrementally construct the set of minimal candidates. Given new measurements the propagation/candidate generation cycle continues until the candidate space has been sufficiently constrained.

⁶The justifications are not shown but are the same as those in Section 3.1.

4 Sequential Diagnosis

In order to reduce the set of remaining candidates the diagnostician must perform measurements [17] which differentiate among the remaining candidates. This section presents a method for choosing a next measurement which best distinguishes the candidates, i.e., that measurement which will, on average, lead to the discovery of the actual candidate in a minimum number of subsequent measurements.

4.1 Possible measurements

The conflict recognition strategy (via P(OBS,ENV)) identifies all predictions for each environment. The results of this analysis provides the basis for a differential diagnosis procedure, allowing GDE to identify possible measurements and their consequences.

Consider how measuring quantity x_i could reduce the candidate space. GDE's data base explicitly represents x_i 's values and their supporting environments:

$$[x_i = v_{ik}, e_{ik1}, \dots, e_{ikm}].$$

If x_i is measured to be v_{ik} , then the supporting environments of any value distinct from the measurement are necessarily conflicts. If v_{ik} is not equal to any of x_i 's predicted values, then every supporting environment for each predicted value of x_i is a conflict. Given GDE's data base, it is simple to identify useful measurements, their possible outcomes, and the conflicts resulting from each outcome. Furthermore, the resulting reduction of the candidate space is easily computed for each outcome.

Consider the example of the previous section. $X = 4$ in environments $\{A_1, M_2\}$ and $\{A_1, A_2, M_3\}$, while $X = 6$ in environment $\{M_1\}$. Measuring X has three possible outcomes: (1) $X = 4$ in which case $\langle M_1 \rangle$ is a conflict and the new minimal candidate is $[M_1]$, (2) $X = 6$ in which case $\langle A_1, M_2 \rangle$ and $\langle A_1, A_2, M_3 \rangle$ are conflicts and the new minimal candidates are $[A_1]$, $[M_2, M_3]$ and $[A_2, M_2]$, or (3) $X \neq 4$ and $X \neq 6$ in which case $\langle M_1 \rangle$, $\langle A_1, M_2 \rangle$ and $\langle A_1, A_2, M_3 \rangle$ are conflicts and $[A_1, M_1]$, $[M_1, M_2, M_3]$ and $[A_2, M_1, M_2]$ are minimal candidates.

The minimal candidates are a computational convenience for representing the entire candidate set. For presentation purposes, in the following we dispense with the idea of minimal candidates and consider all candidates.

The diagnostic process described in the subsequent sections depends critically on manipulating three sets: (1) R_{ik} is the set of (called remaining) candidates that would remain if x_i were measured to be v_{ik} , (2) S_{ik} is the set of (called selected) candidates in which x_i must be v_{ik} (equivalently, the candidates necessarily eliminated if x_i is measured not to be v_{ik}), and (3) U_i is the set of (called uncommitted) candidates which do not predict a value for x_i (equivalently, the candidates which would not be eliminated independent of the value measured for x_i). The set R_{ik} is covered by the sets S_{ik} and U_i :

$$\begin{aligned} R_{ik} &= S_{ik} \cup U_i, \\ S_{ik} \cap U_i &= \phi. \end{aligned}$$

4.2 Lookahead vs. myopic strategies

Section 4.1 describes how to evaluate the consequences of a hypothetical measurement on the set of candidates. By cascading this procedure, we could evaluate the consequences of any sequence of measurements to determine the optimal next measurement (i.e., the one which is expected to isolate the actual candidate in the shortest sequence of measurement). This can be implemented as a classic decision tree analysis, but the computational cost of this analysis is prohibitive. Instead we use a one-step lookahead strategy based on Shannon entropy [2; 23; 29]. Given a particular stage in the diagnostic process we analyze the consequences of each single measurement to determine which one to perform next. To accomplish this we need an evaluation function to determine for each possible outcome of a measurement how difficult it is (i.e., how many additional measurements are necessary) to identify the actual candidate. From decision and information theory we know that a very good cost function is the entropy (H) of the probabilities of the remaining candidates:

$$H = - \sum p_i \log p_i,$$

where p_i is the probability that candidate C_i is the actual candidate given the hypothesized measurement outcome.

Entropy has several important properties (see a reference on information theory [33] for a more rigorous account). If every remaining candidate is equally likely, we have little information to provide discrimination — H is at a maximum. As one candidate becomes much more likely than the rest H approaches a minimum. H estimates the expected cost of identifying the actual candidate as follows. The cost of locating a candidate of probability p_i is proportional to $\log \frac{1}{p_i}$ (cf. binary search through $1/p_i$ objects). The expected cost of identifying the actual candidate is thus proportional to the sum of the product of the probability of each candidate being the actual candidate and the cost of identifying that candidate, i.e., $\sum p_i \log \frac{1}{p_i} = - \sum p_i \log p_i$. Unlikely candidates, although expensive to find, occur infrequently so they contribute little to the cost: $p_i \log \frac{1}{p_i}$ approaches 0 as p_i approaches 0. Conversely, likely candidates, although they occur frequently, are easy to find so contribute little to the cost: $p_i \log \frac{1}{p_i}$ approaches 0 as p_i approaches 1. Locating candidates in between these two extremes is more costly because they occur with significant frequency and the cost of finding them is significant.

4.3 Minimum entropy

Under the assumption that every measurement is of equal cost, the objective of diagnosis is to identify the actual candidate in a minimum number of measurements. This section shows how the entropy cost function presented in the previous section is utilized to choose the best next measurement. As the diagnosis process is sequential, these formulas describe the changes in quantities as a consequence of making a single measurement.

The best measurement is the one which minimizes the expected entropy of candidate probabilities resulting from the measurement. The bulk of this section shows that after substantial algebra and an additional assumption about faulty behavior that the best measurement is the one which minimizes:

$$\sum_{k=1}^m p(x_i = v_{ik}) \log p(x_i = v_{ik}) + p(U_i) \log m,$$

where m is the number of possible values for x_i and,

$$p(S_{ik}) = \sum_{C_j \in S_{ik}} p_j,$$

$$p(U_i) = \sum_{C_j \in U_i} p_j,$$

$$p(x_i = v_{ik}) = p(S_{ik}) + \frac{p(U_i)}{m},$$

v_{i1}, \dots, v_{im} are all possible values⁷ for x_i and $p(x_i = v_{ik})$ is the estimated probability that x_i will be measured to v_{ik} . Note that this formula⁸ can be evaluated directly from the current candidate probabilities.

The equation for $p(x_i = v_{ik})$ is derived as follows. If every candidate predicts a value for x_i , then $p(x_i = v_{ik})$ is the combined probabilities of all the candidates predicting $x_i = v_{ik}$. To the extent that U_i is not empty, the probability $p(x_i = v_{ik})$ can only be approximated with error ϵ_{ik} :

$$p(x_i = v_{ik}) = p(S_{ik}) + \epsilon_{ik}, \quad 0 < \epsilon_{ik} < p(U_i),$$

$$\sum_{k=1}^m \epsilon_{ik} = p(U_i).$$

At any stage of the diagnostic process only some (say the first n of the m possible) of the v_{ik} are actually predicted (i.e., those with non-empty S_{ik}) for x_i . If a candidate does not predict a value for a particular x_i , we assume each possible v_{ik} is equally likely⁹:

$$\epsilon_{ik} = \frac{p(U_i)}{m}.$$

⁷These results can be generalized to account for an infinite number of possible values since, although a quantity may take on an infinite number of possible values, only a finite number of these will be predicted as the consequences of other quantities measured. Furthermore, the entropy resulting from the measurement of a value not predicted is independent of that value. Thus, the system never has to deal with more than a finite set of expected entropies.

⁸If all components fail with equal small probability this equation can be greatly simplified [11].

⁹We could assume that if a component were faulted (i.e., a member of the actual candidate), then its current observed inputs and outputs would be inconsistent with its model. Under such an assumption, the distribution would be skewed away from those v_{ik} predicted from the set of assumptions of the candidate (i.e., viewing a candidate as an environment). We do not make this assumption because a component may appear to be functioning correctly, but actually be faulted producing incorrect outputs for a different set of inputs.

So,

$$p(x_i = v_{ik}) = p(S_{ik}) + \frac{p(U_i)}{m}.$$

Notice that for unpredicted values S_{ik} is empty, so $p(S_{ik}) = 0$ and $p(x_i = v_{ik}) = \frac{p(U_i)}{m}$.

Assuming that the process of taking a measurement doesn't influence the value measured, the expected entropy $H_e(x_i)$ after measuring quantity x_i is given by:

$$H_e(x_i) = \sum_{k=1}^m p(x_i = v_{ik}) H(x_i = v_{ik}).$$

$H(x_i = v_{ik})$ is determined from the candidate probabilities resulting from the hypothesized result $x_i = v_{ik}$. When x_i is measured to be v_{ik} , the probabilities of the candidates shift. Some candidates will be eliminated, reducing their posterior probability to zero. The remaining candidates R_{ik} shift their probabilities according to (see Section 4.4):

$$p'_l = \frac{p_l}{p(x_i = v_{ik})}, \quad l \in S_{ik},$$

$$p'_l = \frac{p_l/m}{p(x_i = v_{ik})}, \quad l \in U_i,$$

where p_l is the current probability of C_l and p'_l is the hypothesized probability. Substituting the candidate probabilities into the entropy equation we obtain:

$$H(x_i = v_{ik}) = - \sum_{l \in R_{ik}} p'_l \log p'_l$$

$$= - \sum_{l \in S_{ik}} \frac{p_l}{p(x_i = v_{ik})} \log \frac{p_l}{p(x_i = v_{ik})}$$

$$- \sum_{l \in U_i} \frac{p_l}{mp(x_i = v_{ik})} \log \frac{p_l}{mp(x_i = v_{ik})}$$

Substituting this into the expected entropy equation we obtain:

$$H_e(x_i) = - \sum_{k=1}^m \sum_{l \in S_{ik}} p_l \log \frac{p_l}{p(x_i = v_{ik})}$$

$$- \sum_{k=1}^m \sum_{l \in U_i} \frac{p_l}{m} \log \frac{p_l}{mp(x_i = v_{ik})}.$$

Expanding the logarithms:

$$H_e(x_i) = - \sum_{k=1}^m \sum_{l \in S_{ik}} p_l \log p_l$$

$$+ \sum_{k=1}^m \sum_{l \in S_{ik}} p_l \log p(x_i = v_{ik})$$

$$- \sum_{k=1}^m \sum_{l \in U_i} \frac{p_l}{m} \log p_l + \sum_{k=1}^m \sum_{l \in U_i} \frac{p_l}{m} \log m$$

$$+ \sum_{k=1}^m \sum_{l \in U_i} \frac{p_l}{m} \log p(x_i = v_{ik}).$$

The first and third terms are simply the current entropy H and is necessarily constant over all x_i :

$$H_e(x_i) = H + \sum_{k=1}^m \sum_{l \in S_{ik}} p_l \log p(x_i = v_{ik}) \\ + \sum_{k=1}^m \sum_{l \in U_i} \frac{p_l}{m} \log m \\ + \sum_{k=1}^m \sum_{l \in U_i} \frac{p_l}{m} \log p(x_i = v_{ik}).$$

Simplifying sums we obtain:

$$H_e(x_i) = H + \sum_{k=1}^m p(S_{ik}) \log p(x_i = v_{ik}) \\ + p(U_i) \log m \\ + \sum_{k=1}^m \frac{p(U_i)}{m} \log p(x_i = v_{ik}).$$

As $p(x_i = v_{ik}) = p(S_{ik}) + \frac{p(U_i)}{m}$:

$$H_e(x_i) = H + \sum_{k=1}^m p(x_i = v_{ik}) \log p(x_i = v_{ik}) + p(U_i) \log m.$$

Thus, the expected entropy can be calculated given the current candidate probability distribution — there is no necessity to explicitly construct the possible posterior probability distributions and compute their entropies. To (usually) obtain positive costs, GDE adds one to the entropy equation. Thus, the best measurement is the one which minimizes:

$$\$(x_i) = \sum_{k=1}^m p(x_i = v_{ik}) \log p(x_i = v_{ik}) + p(U_i) \log m + 1.$$

The choice of base for the logarithm does not affect the relative order of costs. Purely for convenience GDE uses base e (this corresponds to measurements, on average, having e known outcomes). This cost indicates the quality of a hypothesized measurement. The cost is the expected increase in total (i.e., in the entire diagnostic session) number of measurements that need to be made to identify the candidate after making the measurement presuming that measurements have e (admittedly impossible) outcomes. A cost of 1 indicates no information at all is gained because the measurement does not change the entropy. If all measurements had e outcomes, then a cost of 0 would be ideal. However, as the number of outcomes can vary the best cost is $1 - \log m$. Thus, costs can be negative.

4.4 Independence of faults

The initial probabilities of candidates are computed from the initial probabilities of component failure (obtained from their manufacturer or by observation). We

make the assumption that components fail independently. (This approach could be extended to dependent faults except that voluminous data is required.) The initial probability that a particular candidate C_i is the actual candidate C_a is given by:

$$p_i = \prod_{c \in C_i} p(c \in C_a) \prod_{c \notin C_i} (1 - p(c \in C_a)).$$

4.5 The conditional probability of a candidate

Given measurement outcome $x_i = v_{ik}$, the probability of a candidate is computed via Bayes Rule (see Section 6.6):

$$p(C_i | x_i = v_{ik}) = \frac{p(x_i = v_{ik} | C_i) p(C_i)}{p(x_i = v_{ik})}$$

There are three cases for evaluating $p(x_i = v_{ik} | C_i)$. If C_i predicts $x_i = w_{ik}$ where $w_{ik} \neq v_{ik}$ then, the conditional probability is 0:

$$p(x_i = v_{ik} | C_i) = 0 \quad \text{if } C_i \notin R_{ik}.$$

If $w_{ik} = v_{ik}$, then the conditional probability is 1:

$$p(x_i = v_{ik} | C_i) = 1 \quad \text{if } C_i \in S_{ik}.$$

In the third case C_i predicts no value for x_i . We assume that every possible value for x_i (there are m of them) is equally likely:

$$p(x_i = v_{ik} | C_i) = \frac{1}{m} \quad \text{if } C_i \in U_i.$$

Substituting these probabilities into Bayes Rule we obtain:

$$P(C_i | x_i = v_{ik}) = 0 \quad \text{if } C_i \notin R_{ik}, \\ P(C_i | x_i = v_{ik}) = \frac{p(C_i)}{p(x_i = v_{ik})} \quad \text{if } C_i \in S_{ik}, \\ P(C_i | x_i = v_{ik}) = \frac{p(C_i)/m}{p(x_i = v_{ik})} \quad \text{if } C_i \in U_i,$$

where

$$p(x_i = v_{ik}) = p(S_{ik}) + \frac{p(U_i)}{m}.$$

The estimate $p(x_i = v_{ik} | C_i) = 1/m$ is suspect and introduces various computational complexities. Fortunately, U_i contains primarily low probability candidates and thus any error tends to be minor. On average, the candidates in U_i are non-minimal (because minimal candidates tend to assign values to most of the device's variables) and as non-minimal candidates have lower probability than minimal candidates, the candidates in U_i have relatively lower probability. In practice, using $P(x_i = v_{ik} | C_i) = 1$ for $C_i \in R_{ik}$ greatly simplifies the computations and rarely affects the number of measurements required.

4.6 Examples

This section presents a series of example to illustrate some of the intuitions behind our technique.

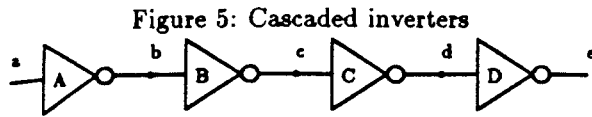


Table 1: Expected costs for cascaded inverters after measurements (with $p = .01$)

	a=1	a=1,e=0	a=1,e=1
a	1	1	1
b	.98	.45	.999
c	.96	.31	.998
d	.94	.44	.999
e	.93	1	1

4.6.1 Cascaded inverters: $a = 1$

Consider circuit of Figure 5. Assume the initial probability of component failure is .01 and input $a = 1$ is given. As initially no symptom has been detected, on average little is gained by making any measurement—all costs are nearly 1. Compare the advantages and disadvantages of measuring b or e . If the value measured differs from the value predicted, then measuring closer to the input (or any previously measured value) produces a smaller conflict, and thus more information. For example, if we measure b to be 1, then we learn that A must be faulted, whereas measuring $e = 0$ tells us only that some component is faulty. On the other hand, measuring further away from the input is more likely to produce a discrepant value. That is, as there are a large number of intervening components, it is more likely that one will be faulted and thus produce an incorrect value. These two effects tend to cancel each other, however, the probability of finding a particular value outweighs the expected cost of isolating the candidate from a set. In terms of entropy, every possible outcome contributes $p_i \log \frac{1}{p_i}$ but p_i dominates $\log \frac{1}{p_i}$. In general for any cascaded sequence of components for which only the input is known, the best next measurement is the output of the sequence. These expected costs are enumerated under column headed $a = 1$ of Table 1.

4.6.2 Cascaded inverters: $a = 1, e = 0$

Suppose e is measured to be 0, violating the prediction and thus telling us that at least one of the components is faulted. The best next measurement is the one that is equidistant from the previous two measurements, c (see column $a = 1, e = 0$ of Table 1).

4.6.3 Cascaded inverters: $a = 1, e = 1$

Suppose that instead of measuring e to be 0 it was measured 1 as predicted. The circuit must have a relatively unlikely double fault (if it has any fault at all) where one fault shadows the effect of the other. Nevertheless the best measurement to identify a possible double fault is still c (see column $a = 1, e = 1$ of Table 1).

Table 2: Expected costs for cascaded inverters after measurements (with $p(B, C, D) = .01, p(A) = .025$)

	a=1,e=1
a	1
b	.33
c	.36
d	.53
e	1

Table 3: Expected costs for Figure 2 after measurements.

	initial	F=10	G=12	X=6	Y=4
F	.88	1	1	1	1
G	.88	.28	1	1	1
X	.95	.34	.28	1	1
Y	.95	.34	.94	.90	1
Z	.95	.95	.97	.94	.141

4.6.4 Cascaded inverters: $a = 1, e = 1, p(A) = .025$

If we are given that the first inverter A is more likely to be faulted, then the best measurement point is no longer equidistant from the previous observations. The advantage of measuring b first instead of c is that we might identify that inverter A is faulted immediately. As A is far more likely to be faulted, it is better to measure b first (see Table 2).

4.6.5 Diagnosing a fault

Consider the circuit of Figure 2. Suppose that M_2 is faulted exhibiting behavior $Y = B \times C - 2$ and A_2 is faulted with behavior $G = Y \times Z + 2$. The inputs are the same as earlier. The following illustrates GDE's strategy to localize this fault. Note that this is an unusual fault in that the effect of the two faults cancel, producing the value predicted for G . Thus, several measurements are necessary to identify the fault. We assume that component's fail with initial probability .01 and that $m = 16$).

Initially the most probable candidate is \emptyset , $p(\emptyset = C_a) = .951$. The expected cost associated with each hypothetical measurement is given in the second column of Table 3. All the costs are nearly one, indicating that little is to be gained by any of the measurements, this is because on average components are not faulted, and there is yet no evidence that anything is malfunctioning. Not surprisingly, F and G as well as X, Y, Z are treated symmetrically. F and G have slightly lower cost because their predicted values depend on three components functioning correctly, while $X, Y,$ and Z depend on two (i.e., F and G are more likely to have discrepant values).

Suppose F is measured with result $F = 10$. The most probable candidates are now $[A_1], [M_1],$ and $[M_2]$, all

having probability .323. The expected costs are given by the third column of Table 3. G , X and Y are all good measurements because each one differentiates among the high-probability single-fault candidates. Each high-probability candidate predicts $Z = 6$ so measuring Z provides little new information. G is a slightly better point to measure because the candidates are more balanced between G 's two predicted outcomes. (the best measurement is one whose predicted outcomes all have equal probability and which cover all the candidates).

Next, suppose G is measured with result $G = 12$. The most probable candidates are now $[A_1]$ and $[M_1]$ both with probability 0.478. The expected costs are given by the fourth column of Table 3. At this point X is the best measurement because it splits the two high-probability candidates.

Next, suppose X is measured with result $X = 6$. This results in a single high-probability candidate $[A_1]$ with probability .942. The next seven most likely candidates are: $[A_1M_3]$, $[A_1M_2]$, $[A_1M_1]$, $[A_1A_2]$, $[A_2M_2]$, and $[M_2M_3]$, all with probability 0.00951. The expected costs are given by the fifth column of Table 3.

Next, suppose Y is measured with result $Y = 4$. M_2 is now necessarily faulted and at least one other fault exists. The expected costs are given by column $Y = 4$ of Table 3.

Finally, suppose Z is measured with result $Z = 6$. There are six remaining candidates: $[A_2M_2]$ with probability 0.970, $[A_2M_1M_2]$, $[A_1A_2M_2]$, $[A_2M_2M_3]$, with probabilities 0.0098, $[A_1A_2M_1M_2]$, $[A_1A_2M_2M_3]$, $[A_2M_1M_2M_3]$, with probabilities 0.0001, and $[A_1A_2M_1M_2M_3]$ with probability 0.000001. Components M_2 and A_2 are necessarily faulted, and A_1 , M_1 , and M_3 are possibly faulted, each with probability .01. No measurement points remain in the circuit, so no further information can be obtained.

4.7 Logical incompleteness

In practice, for diverse reasons, the underlying inference process is usually incomplete. One of the consequences of this incompleteness is that it becomes more difficult to evaluate the results of hypothetical measurements — the U_i are larger than ideal. As any incompleteness degrades GDE's performance, it is instructive to examine the sources and types of this incompleteness.

An often avoidable form of incompleteness occurs when the conflict recognition strategy misses some conflicts. This was discussed earlier. Here we assume the conflict recognition strategy is complete. U_i will be larger than ideal only if the prediction process is incomplete. This incompleteness manifests itself in two ways. First, an incomplete inference process may result in missing predicted values and missing supporting environments. As a consequence, the hypothesized conflicts resulting from measuring a quantity will be incomplete. For example, consider an inverter which is incompletely modeled by a rule which predicts its output from its input, but not its input from its output. Even though the inverter's output is measured to be one, the prediction

that its input is zero is not made. Thus, GDE does not consider measuring its input. A second source of incompleteness is inherent in the $x_i = v_{ik}$ representation — there may be many additional properties that could be inferred about x_i , but as there is no way to represent them they cannot be used by our strategy. For example, it cannot represent $x \neq 1$. Thus, although $x \neq 1$ might be derivable from an environment, GDE cannot foresee the resulting conflict when considering, say, $x = 2$.

Assuming the basic conflict recognition strategy is complete, both these sources of incompleteness can be avoided at prohibitive computational costs. The first source of incompleteness can be avoided with a complete inference process. The second source of incompleteness can be avoided with a more general representation¹⁰.

5 Pragmatics

5.1 Most probable candidates

Computing all candidates is computationally prohibitive. In practice it is only necessary to compute the more probable candidates¹¹. There is no way to tell whether a single candidate *actually* has a high probability without knowing the overall normalization factor. This suggests using a best-first search of the lattice to find candidates in decreasing probability order. This search is arbitrarily stopped for candidates below some threshold fraction (e.g., $\frac{1}{10}$ of the highest probability candidate). Although the most probable candidate is a minimal candidate, the remaining minimal candidates need not be very probable.

5.2 When to stop making measurements

If there are many points in a device that could be measured, then choosing the best point to measure can be computationally expensive. A heuristic is to make the first reasonable measurement whose cost is computed to be less than $1 - \log .5 = .7$ as this measurement on average, splits the candidate space in half.

The point at which measurement should stop depends greatly on the seriousness of a misdiagnosis as well as on the a priori probabilities of component failure. When a candidate is found whose probability approaches some threshold (e.g., .9), diagnosis can stop. If the cost of misdiagnosis is high, then the threshold should be increased.

6 Comparison to Other Work

6.1 Circuit diagnosis based on structure and behavior

Work in the AI community on model-based hardware diagnosis has grown out of a desire to move away from

¹⁰ Assuming the number of possible values for quantities are finite, GDE could hypothesize each possible measurement outcome, run its complete inference procedure and precisely compute the R_{ik} from which the S_{ik} could then be computed.

¹¹ These issues are addressed in far more depth in [12].

the domain and device specific fault models used in traditional circuit diagnosis.¹² Instead, during candidate generation the model-based approach reasons from a small set of component behavior models plus the structure of the device. The requirement for device specific fault models is eliminated by basing the diagnostic approach solely on the knowledge that, if a component's behavior is inconsistent with its model, then it must be faulty. This results in a domain independent diagnostic technique. A number of systems have followed this approach for diagnosing both analog (e.g., LOCAL [10] and SOPHIE [4]) and digital circuits (e.g., HT [6] and DART [15]).

Our work naturally extends this approach along a number of dimensions. First, unlike earlier approaches, our work is aimed specifically at coping with the problem of diagnosing multiple faults. Earlier work focused primarily on the case where all symptoms could be explained by a single component being faulty. As Davis points out, the obvious extension to his work to handle multiple faults results in an algorithm which is exponential in the number of potential faults. As our approach represents the candidate space implicitly in terms of the minimal candidates, we need only be concerned with the growth of this smaller set. Typically the size of each minimal candidate is relatively small (i.e., the symptom can usually be explained by one, two or three components being simultaneously faulty), thus in practice our approach tends to grow with the square or cube of the number of potential faults.

Second, our approach to diagnosis is inference procedure independent, as well as domain independent. LOCAL, SOPHIE, and Davis' system all represent circuit knowledge as constraints and then use some form of constraint propagation to infer circuit quantities and their dependencies. The motivation behind this approach is that constraints naturally reflect the local interaction of behavior in the real world. These are similar (although not identical) to the constraint propagation technique used to demonstrate GDE in this paper.

On the other hand, DART expresses circuit knowledge as logical propositions and uses an inference system based on resolution residue to infer circuit properties. This same general inference procedure allows DART to deduce which components are suspect and how to change circuit inputs to reduce the suspect set to a singleton. Such a resolution-based inference strategy can be incorporated into the candidate generation and conflict recognition portion of GDE by recording, for each resolution step, the dependence of the resolvent on the formulas used to perform the resolution. Then, as in the case for constraint propagation, GDE can guide the inference process in identifying minimal candidates first.

Resolution is highlighted as having an advantage over constraint propagation in that it is logically complete

¹²See Davis [6], sections 5.1., 12.1. and 13 for a discussion of traditional circuit diagnosis and the advantages of the model-based approach. Also see [19].

with respect to a first order theory, where most constraint propagators are not. However, this can be misleading. A significant computational cost is incurred using a logically complete inference strategy, such as resolution. Yet a logically complete inference strategy does not guarantee that the set of predictions will also be complete. For example in an analog domain, producing exact predictions often involves solving systems of higher order non-linear differential equations. As this type of equation is not generally solvable with known techniques, completeness in the predictor is currently beyond our reach. In practice propagation of constraints (without symbolic algebra) provides a good compromise between completeness and computational expense.

Third, our approach is incremental. This is crucial as diagnosis is an iterative process of making observations, refining hypotheses, and then using this new knowledge to guide further observations.

Finally, our approach is unique in that it combines model-based reasoning with probabilistic information in a sequential probing strategy. SOPHIE also proposes measurements to localize the failing component, but is based on an ad-hoc half-split method. Its design is based on the presupposition that the circuit contains a single fault. Thus, candidate generation is trivial (set intersection), and identifying good measurements is easy.

6.1.1 Possible extensions

Representing and manipulating candidates in terms of minimal candidates gives us a significant computational advantage over previous approaches; however, even this representation can grow exponentially in the worst case. To effectively cope with very large devices, additional techniques must be incorporated.

One approach involves grouping components into larger modules and modifying the candidate generation strategy to deal with hierarchical decompositions. Although we have not implemented this, the strategies proposed by Davis and Genesereth, as they are model-based, apply. Their basic idea is to troubleshoot at the most abstract level in the hierarchy first and only analyze the contents of a module when there is reasonable evidence to suspect it. If there is no conflict involving the module, then there is no reason to suspect it.

Complexity is directly related to the number of hypothetical faults entertained. Thus another approach, proposed by Davis, involves enumerating and layering categories of failures based on their likelihood. Troubleshooting begins by considering faults in the most likely category first, moving to less likely categories if these fail to explain the symptoms. GDE uses a variation on this approach whereby candidates can be enumerated based on their likelihood. However, no theory has been developed about failure categories.

LOCAL and SOPHIE also address a number of issues which GDE does not. LOCAL and SOPHIE use corroborations as well as conflicts to eliminate candidates. To take advantage of corroborations, SOPHIE includes an *exhaustive* set of fault modes for each component.

As a consequence, SOPHIE can identify a component as being unfaulted by determining that it is not operating in any of its fault modes (i.e., none of the components fault modes can explain the failure, and we know all the component's modes of failure, so the component must be working). Thus, SOPHIE reduces the candidate space from both sides: conflicts eliminate candidates that do not explain a conflict, and corroborations eliminate candidates which include known working components.

In addition, SOPHIE and LOCAL dealt with the problem of imprecise models and values as well. This makes constraint propagation very difficult because it is hard to tell whether two differing values corroborate or conflict within the precision of the analysis.

GDE proposes optimal measurements given a fixed set of inputs. Both Shirley's system [34] and DART generate test vectors to localize circuit faults. This approach is useful in cases where it is both difficult to make internal measurements, and easy to change circuit inputs. Both approaches produce tests which are likely to give useful information; however, no serious attempt is made at selecting the optimal tests. A fairly simple extension can be made to our cost function to account for tests, by taking into consideration, for example the cost of modifying the inputs.

In Shirley's approach, each test isolates a single potentially faulted component by using known good components to guide the input signals to the faulty component and the output of that component to a point which can be measured. However, it is not always possible to route the test signals such that all other suspects are avoided. Using the information constructed by GDE, this approach can be modified, so that tests are constructed first for components which are most likely to be faulted, while propagating the signal along a path with the lowest probability of having a faulty component.

6.2 Counterfactuals

Ginsberg [16] points out how David Lewis' possible world semantics could be applied to diagnosis. Each component is modeled by a counterfactual, a statement such as "if p , then q ," where p is expected to be false. Thus A_1 is modeled by the counterfactual "if A_1 fails, then $F = X + Y$ need not hold." Counterfactuals are evaluated by considering a "possible world" that is as similar as possible to the real world as possible where p is true. The counterfactual is true if q holds in such a possible world. In this framework the most similar worlds (i.e., those closest to the one in which the circuit function correctly) correspond directly to our minimal candidates. A most similar world is one in which as few of the p 's (e.g., " A_1 fails") as possible become true. Thus, in this instance Ginsberg uses counterfactuals for hypotheticals.

Ginsberg's approaches handles multiple faults, but he does not discuss measurement strategies or probabilities. Both could be integrated into his approach.

6.3 Reiter

Reiter¹³ [32] has been independently exploring many of the ideas incorporated in GDE. His theory of diagnosis provides a formal account of our "intuitive" techniques for conflict recognition and candidate generation. However, his theory does not include a theory of measurements nor how to exploit probabilistic information.

Reiter's theory uses McCarthy's [21] AB predicate. Reiter writes $\neg AB(A_1)$ (i.e., adder A_1 is not AB normal) while we write A_1 (i.e., the assumption that adder A_1 is working correctly). Under this mapping, Reiter's definition of diagnosis is equivalent to our definition of minimal candidate, and his definition of conflict set is equivalent to our definition of conflict.

Reiter proposes (unimplemented) a diagnostic algorithm based on his theory. This architecture is quite different from GDE's. The theorem prover of his architecture corresponds roughly to our inference engine. While GDE's procedure first computes all minimal conflicts resulting from a new measurement before updating candidate space, Reiter's algorithm intermingles conflict recognition with candidate generation to guide the theorem proving to "prevent the computation of inessential variants of refutations, without imposing any constraints on the nature of the underlying theorem proving system." Given GDE's inference architecture (and the necessity to choose the best next measurement), GDE avoids almost all these inessential variants and also avoids much of the computation Reiter's architecture demands.

6.4 SNePS

SNePS [20] incorporates a belief revision system SNeBR which reasons about the consistency of hypotheses. This belief revision system has been [7] applied to fault detection in circuits. Some of the basic concepts used in the SNePS approach are similar to those used in GDE — both use an assumption-based belief revision system (see [9] for a comparison). As described in [7] SNePS is weaker than either GDE or Reiter's approach. In GDE's terminology, SNePS operates in a single environment, detecting symptoms (and hence conflicts) for only those variables about which a user query is made. The system provides no mechanisms to detect all symptoms (and thus all conflicts), to identify candidates, to propose measurements, or to exploit probabilistic information. However, there seems no reason, in principle, that the methods of GDE cannot be incorporated into their system.

6.5 IN-ATE and FIS

IN-ATE [5] and FIS [27] are expert system shells specifically designed for fault diagnosis. Although neither is purely model-based (i.e., not purely within the structure-function paradigm), they are the only other electronic diagnosis systems we are aware of that pro-

¹³These results are generalized in [13].

pose measurements, and thus it is instructive to compare them with GDE.

IN-ATE incorporates two formal criteria to determine the best next measurement: minimum entropy (as in GDE), and gamma miniaverage heuristic search [35]. We chose minimum entropy (as FIS does) in GDE because it was the best evaluation function which is efficiently computable (i.e., it is based on one-step lookahead). The gamma miniaverage heuristic search is a multi-step lookahead procedure and hence computationally expensive to apply. Nevertheless, there is no reason, in principle, it could not be incorporated into GDE.

Unlike GDE, IN-ATE and FIS can exploit a variety of kinds of knowledge (e.g., fault trees, expert-supplied rules, hierarchy) in the diagnostic process. However, viewed from the pure model-based approach, they are limited in predictive power and as a consequence can only estimate the probabilities for their predictions.

As IN-ATE is not model-based, it does not construct explicit value predictions as does GDE. Instead, it predicts whether a particular measurement outcome will be "good" or "bad." (This, in itself, presents difficulties because a measurement being "good" or "bad" is relative to the observations being compared to. However, IN-ATE takes "good" and "bad" as absolute.) These predictions are computed by analyzing expert-supplied empirical associations based on "good" and "bad." IN-ATE incorporates a heuristic rule generator which exploits connectivity information, but it, at a minimum, must be provided with the definitions of "good" and "bad" for the test points. In this approach, the number of test points is limited to those explicitly mentioned in rules (or for which "good" and "bad" are externally defined). With this limited set of test points full search (i.e., gamma miniaverage search) becomes plausible. FIS exploits more model knowledge, propagating qualitative values such as {hi,ok,lo} through the circuit topology via expert-supplied causal rules. The propagated values are always relative to fixed expert-supplied values. Thus neither FIS nor IN-ATE can analyze hypothetical faults.

All three systems must compute probabilities for their predictions. GDE directly computes the probability of the predictions from the probabilities of the candidates involved. IN-ATE has a much more difficult time determining good/bad probabilities because it cannot determine the candidate space and their associated probabilities. Instead it uses the expert-supplied rules to propagate probabilities (much like MYCIN) and Dempster-Shafer to combine the resulting evidence. FIS' definition of candidate probability is similar to GDE's, however, as it cannot perform hypothetical reasoning it can only estimate the probabilities of its predictions.

6.6 Medical diagnosis

There has been extensive research on the diagnostic task in the medical decision making community. Although most of this research is not model-based, its concern with identifying the diseases causing symptoms and subse-

quently proposing information-gathering queries is analogous to ours.

Like Gorry and Barnett [17] GDE solves a sequential diagnostic task. The fundamental equation (Bayes Rule) (formula (2) of [38]) states that the posterior probability of hypothesis H_i being the true state H_T given evidence Q_i is:

$$p(H_j|Q_i) = \frac{p(Q_i|H_j)p(H_j)}{\sum_{k=1}^n p(Q_i|H_k)p(H_k)},$$

where $p(Q_i|H_j)$ is the conditional probability of evidence Q_i while $p(H_j)$ is the a priori probability of H_j . Given these probabilities it is possible to determine the posterior probability distribution, and, in addition, to determine the hypothetical probability distribution resulting from proposed tests. Thus, this provides complete information to evaluate the expected gain of a test [17] and to select the best test to make next.

The practical obstacle to employing Bayes rule is the unavailability of the conditional probabilities. These probabilities are both hard to estimate and extremely numerous. For even small numbers of possible symptoms and diseases the number of probabilities required is extremely large (given just 10 hypotheses and 5 possible binary tests, 2420 conditional probabilities are required [38]).

The engineering domain differs from the medical domain in that very accurate models exist based on the structure of the (faulty) system. Thus, exploiting model-based reasoning, GDE computes these conditional probabilities directly rather than depending on empirical results. Another advantage of the model-based approach for engineering is that the potential tests are computed in the course of the analysis instead of having them supplied a priori.

Early research [17] presumed that the patient had only one disease. This is analogous to the single-fault assumption in circuit diagnosis. More recent research allows for multiple simultaneous diseases. Bayes rule can also be applied to this multimembership classification problem [2; 3] but the number of conditional probabilities required becomes exponentially larger than the single-disease case (which already is a large number). Again, GDE computes these conditional probabilities by model-based reasoning.

The concept of set covering used in [24; 25; 26; 28; 30; 31] is similar to the candidate generation phase of our diagnostic approach. The irredundant set cover of the General Set Covering theory (GSC) of Reggia corresponds directly to our notion of minimal candidates. The primary difference is that we consider, in theory, every possible cover to be an explanation for the symptoms while GSC considers only the minimal candidates as explanations. Although GSC incorporates a crude heuristic strategy for proposing new measurements, as the conditional probabilities are available to GDE, the preferred approach, minimum entropy, can be applied.

7 ACKNOWLEDGMENTS

Daniel G. Bobrow, Randy Davis, Kenneth Forbus, Matthew Ginsberg, Frank Halasz, Walter Hamscher, Tad Hogg, Ramesh Patil, Leah Ruby, Olivier Raiman, Mark Shirley and Jeff Van Baalen, provided useful insights. We especially thank Ray Reiter for his clear perspective and many productive interactions. Lenore Johnson and Denise Pawson helped edit drafts and prepare figures.

References

- [1] Ben-Bassat, M., Myopic policies in sequential classification, *IEEE Transactions on computers* C-27 (1978) 170-178.
- [2] Ben-Bassat, M., Multimembership and multiperspective classification: introduction, applications, and a Bayesian model, *IEEE Transactions on System, Man and Cybernetics*, 6 (1980) 331-336.
- [3] Ben-Bassat, Moshe, R.W. Carlson, V.K. Puri, Mark D. Davenport, J.A. Schriver, Mohamed Latif, Ronald Smith, Larry D. Portigal, Edward H. Lipnick, and Max Harry Weil, Pattern-based interactive diagnosis of multiple disorders: The MEDAS system, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2 (1980) 148-160.
- [4] Brown, J.S., Burton, R. R. and de Kleer, J., Pedagogical, natural language and knowledge engineering techniques in SOPHIE I, II and III, in: D. Sleeman and J.S. Brown (Eds.), *Intelligent Tutoring Systems*, (Academic Press, New York, 1982) 227-282. An expansion of the relevant sections of this paper appears in these readings under the title, Model-based diagnosis in SOPHIE III.
- [5] Cantone, R.R., Lander, W.B., Marrone, M.P., and Gaynor, M.W., IN-ATE: Fault diagnosis as expert system guided search, in: L. Bole and M.J. Coombs (Eds.), (Springer-Verlag, New York, 1986).
- [6] Davis, R., Diagnostic Reasoning based on structure and behavior, *Artificial Intelligence* 24 (1984) 347-410. This paper appears in these readings.
- [7] Campbell, S.S., and Shapiro, S.C., Using belief revision to detect faults in circuits, SNeRG Technical Note No. 15, Department of Computer Science, SUNY at Buffalo, 1986.
- [8] de Kleer, J., An assumption-based truth maintenance system, *Artificial Intelligence* 28 (1986) 127-162. Also in *Readings in NonMonotonic Reasoning*, edited by Matthew L. Ginsberg, (Morgan Kaufmann, 1987), 280-297.
- [9] de Kleer, J., Problem solving with the ATMS, *Artificial Intelligence* 28 (1986) 197-224.
- [10] de Kleer, J., Local methods of localizing faults in electronic circuits, *Artificial Intelligence Laboratory, AIM-394*, Cambridge: M.I.T., 1976. Sections of this paper appear in these readings in a paper with the title, Model-based diagnosis in SOPHIE III.
- [11] de Kleer, J., Using crude probability estimates to guide diagnosis, *Artificial Intelligence* 45 (1990) 381-391. This paper appears in these readings.
- [12] de Kleer, J. and Williams, B.C., Diagnosis with behavioral modes, in: *Proceedings IJCAI-89*, Detroit, MI (1989) 104-109. A version of this paper appears in these readings.
- [13] de Kleer, J., Mackworth A., and Reiter R., Characterizing Diagnoses and Systems, *Artificial Intelligence* 56 (1992). This paper appears in these readings.
- [14] Doyle, J., A truth maintenance system, *Artificial Intelligence* 12 (1979) 231-272.
- [15] Genesereth, M.R., The use of design descriptions in automated diagnosis, *Artificial Intelligence* 24 (1984) 411-436. This paper appears in these readings.
- [16] Ginsberg, M.L., Counterfactuals, in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, Los Angeles, CA (1985) 107-110.
- [17] Gorry, G.A., and Barnett, G.O., Experience with a model of sequential diagnosis, *Computers and Biomedical Research* 1 (1968) 490-507.
- [18] Hamscher, W., and Davis, R., Diagnosing circuits with state: an inherently underconstrained problem, in: *Proceedings of the National Conference on Artificial Intelligence*, Austin, TX (August, 1984) 142-147. This paper appears in these readings.
- [19] Davis, R., and Hamscher, W., Model-based reasoning: Troubleshooting, in *Exploring artificial intelligence*, edited by H.E. Shrobe and the American Association for Artificial Intelligence, (Morgan Kaufmann, 1988), 297-346. This paper appears in these readings.
- [20] Martins, J.P. and S.C. Shapiro, Reasoning in multiple belief spaces, *Proceedings of the International Joint Conference on Artificial Intelligence*, 1983.
- [21] McCarthy, J., Applications of circumscription to formalizing commonsense knowledge, *Artificial Intelligence* 28 (1986) 89-116.
- [22] Mitchell, T., Version spaces: An approach to concept learning, Computer Science Department, STAN-CS-78-711, Palo Alto: Stanford University, 1978.
- [23] Pearl, J., Entropy, information and rational decisions, *Policy analysis and information systems*, 3 (1979) 93-109.
- [24] Peng, Y.P. and Reggia, J.A., Plausibility of diagnostic hypotheses: The nature of simplicity, in: *Proceedings AAAI-86*, Philadelphia, PA (August, 1986), 140-145.
- [25] Peng, Y.P. and Reggia, J.A., A probabilistic causal model for diagnostic problem-solving; part one: Integrating symbolic causal inference with numeric

- probabilistic inference, Department of Computer Science, University of Maryland, 1986.
- [26] Peng, Y.P. and Reggia, J.A., A probabilistic causal model for diagnostic problem-solving; part two: Diagnostic strategy, Department of Computer Science, University of Maryland, 1986.
 - [27] Pipitone, F., The FIS electronics troubleshooting system, *IEEE Computer* 19, (July 1986) 68-75.
 - [28] Pople, H., The formation of composite hypotheses in diagnostic problem solving: an exercise in synthetic reasoning, *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Pittsburg, PA (August, 1977), 1030-1037.
 - [29] Quinlan, J.R., Learning efficient classification procedures and their application to chess end games, in: *Machine Learning*, edited by R.S. Michalski, J.G. Carbonell and T.M. Mitchell, (Tioga, Palo Alto, CA, 1983), 463-482.
 - [30] J.A. Reggia, and Nau, D.S., An abductive non-monotonic logic, *Workshop on non-monotonic reasoning*, (October, 1984).
 - [31] J.A. Reggia, Nau, D.S., and Wang, P.Y., Diagnostic expert systems based on a set covering model, *International Journal of Man-Machine Studies* 19 (1983) 437-460.
 - [32] Reiter, R., A theory of diagnosis from first principles, *Artificial Intelligence* 32 (1987) 97-130. This paper appears in these readings.
 - [33] Shannon, C.E., A mathematical theory of communication, *Bell System Technical Journal* 27 (1948) 379-623.
 - [34] Shirley, M.H. and Davis R., Generating Distinguishing Tests Based on Hierarchical Models and Symptom Information, *Proceedings IEEE International Conference on Computer Design*, Oct. 1983.
 - [35] Slagle, J.R., and Lee, R.C.T., Application of game tree searching techniques to sequential pattern recognition, *Communications of the ACM* 14 (1971) 103-110.
 - [36] Steele, G.L., The definition and implementation of a computer programming language based on constraints, AI Technical Report 595, MIT, Cambridge, MA, 1979.
 - [37] Sussman, G.J. and Steele, G.L., CONSTRAINTS: A language for expressing almost-hierarchical descriptions, *Artificial Intelligence* 14 (1980) 1-39.
 - [38] Szolovits, P. and Pauker, S.G., Categorical and probabilistic reasoning in medical diagnosis, *Artificial Intelligence* 11 (1978) 115-144.
 - [39] Williams, B.C., Doing time: Putting qualitative reasoning on firmer ground, *Proceedings of the National Conference on Artificial Intelligence*, Philadelphia, PA, (August, 1986), 105-112.

DIAGNOSIS WITH BEHAVIORAL MODES¹

Johan de Kleer
Brian C. Williams
Xerox Palo Alto Research Center
3333 Coyote Hill Road, Palo Alto CA 94304
Email: dekleer, bwilliams@parc.xerox.com

Abstract

Diagnostic tasks involve identifying faulty components from observations of symptomatic device behavior. This paper presents a general diagnostic theory that uses the perspective of diagnosis as *identifying consistent modes of behavior*, correct or faulty. Our theory draws on the intuitions behind recent diagnostic theories to identify faulty components without necessarily knowing how they fail. To derive additional diagnostic discrimination we use the models for behavioral modes together with probabilistic information about the likelihood of each mode of behavior.

1 Introduction

When you have eliminated the impossible, whatever remains, *however improbable*, must be the truth. — *Sherlock Holmes. The Sign of the Four.*

The objective of our research is to develop a general theory of diagnosis that captures a human diagnostician's predominant modes of reasoning. This theory is intended to serve as the conceptual foundation for computational systems that diagnose devices.

Early approaches [1, 4] to diagnosis used fault models to identify failure modes of faulty components that explain the observations made. The ability to predict failing components' behaviors provided powerful diagnostic discrimination. However, these techniques depend on the assumption that all failure modes are known *a priori* — an assumption that is sometimes warranted but is never guaranteed. The unacceptable result of not satisfying this assumption — *faulty diagnoses* — has led researchers to abandon this powerful approach.

The model-based diagnostic approach adopted by most recent researchers [3, 6, 9] provides a framework for diagnosing a device from correct behavior only. This approach is based on the observation that it is not necessary to determine how a component is failing to know that it is faulty — a component is faulty if its correct behavior (i.e., as specified by its manufacturer) is inconsistent with the observations. Since only correct behavior needs to be modeled, any knowledge about the behavior of component

fault modes is ignored. This provides a fundamental advantage over earlier techniques requiring *a priori* knowledge of all fault modes. Unforeseen failure modes pose no difficulty. However, what is lost is the additional diagnostic discrimination derived from knowing the *likely* ways a component fails, and the ability to determine whether these failure modes are consistent with the observations. Thus, unlikely possibilities are entertained as seriously as likely ones. For example, as far as most model-based diagnostic approaches are concerned, a light bulb is equally likely to burn out as to become permanently lit (even if electrically disconnected).

Human diagnosticians, however, take great advantage of behavioral models of known failure modes, together with the likelihood that these modes will occur. Knowledge of fault modes is used to pinpoint faulty components faster, and to help determine specific repairs that must be made to the faulty components.

We view the central task of diagnosis as identifying the behavioral modes (correct or faulty) of all the components. Whether a mode is faulty or not is irrelevant. Our synthesis hypothesizes that it is not the notion of fault, but behavioral mode that is fundamental to diagnosis. Each component has a set of possible behavioral modes including an unknown mode which makes no predictions, and therefore can never conflict with the evidence. The unknown mode is included to allow for the possibility, albeit small, of unforeseen behavioral modes. This unknown mode is crucial because early diagnostic algorithms, when confronted with an unforeseen fault mode, either start making useless probes or simply give up. Our approach pinpoints the failing component as behaving in an unknown mode.

The introduction of fault models potentially introduces significant computational overhead for the diagnostic algorithms. Diagnosing multiple faults is inherently a combinatoric process. Introducing fault models exacerbates the process, by introducing multiple modes and possible behaviors to consider. To control the combinatorics we introduce computational techniques which focus reasoning on more probable possibilities first. These techniques, in effect, focus diagnostic reasoning only on those component behavioral modes that are more probable given the evidence. This set grows and shrinks as evidence is collected.

By using the new perspective of diagnosis as identifying probable behavioral modes, we are able to extend our earlier work on model-based diagnosis (the General Diagnostic Engine (GDE) [6]) to reason about modes of behavior. The resulting system we call *Sherlock*. GDE provides a general domain-independent architecture for diag-

¹This paper originally appeared in *Proceedings IJCAI-89*, Detroit, MI (1989) 104-109.

nosing any number of simultaneous faults in a device given solely a description of its structure (e.g., electrical circuit schematic) and specifications of correct component behaviors (e.g., that resistors obey Ohm's law). Given a set of observations, GDE constructs hypotheses (called *diagnoses*) identifies the faulty components and suggests points where additional measurements (called *probes*) should be made to localize the diagnosis with as few measurements as possible.

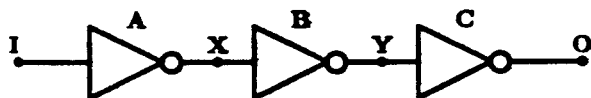
We have implemented our approach by extending GDE to Sherlock, and have tested it on a variety of digital circuits — from simple three inverter circuits, to ALUs consisting of 400 gates with 4 behavioral modes each. Sherlock exploits knowledge of failure modes to pinpoint faults more equally and identify in what mode components are functioning.

2 Related work

Exploiting the use of fault models has recently become an active research area [10, 11, 12, 15]. In particular, Holtzblatt's [11] generalization of GDE incorporates the notion of behavioral modes in a similar spirit to Sherlock. But Holtzblatt's GMODS system is missing many key features of Sherlock such as accommodating unexpected failures, incorporating probabilistic information to rank diagnoses and guide probing, incorporating most-probable-first heuristics to limit the computational complexity which arises for larger devices, and combining evidence gathered from multiple observations of a device. As GMODS does not use probabilistic information it relies on an expensive hyperresolution rule to rule out fault modes and cannot focus reasoning on more probable diagnoses. Struss [15] argues against the use of probabilistic information and the use of an unknown mode. Instead he employs a resolution rule and controls reasoning to introduce appropriate fault modes only when necessary. Through the use of an alternative architecture which redefines the notion of fault, Raiman [12] achieves some of the advantages of knowledge of fault modes without having to incorporate them. Hamscher [10] incorporates fault models with his generalization of GDE called XDE.

3 Diagnosis with modes

The perspective of diagnosis as identifying probable behavioral modes is best appreciated through an example. Consider the simple three inverter circuit shown below. Suppose that the input (I) is set to zero, and that, al-



though the output (O) should be one if functioning correctly, it is measured to be zero. Without knowledge of fault modes, all three inverters are equally likely to be faulted. If we knew that inverters (almost) always failed with output stuck-at-1, then we could infer that inverter B was likely to be faulted. Thus knowledge of failure modes can provide significant diagnostic information.

Knowledge of failure modes is also important to decide what measurement to make next. If all faults were equally likely, measuring X or Y provides equal information. However, suppose we know that inverters A and B almost always fail by having their output stuck-at-1, and that inverter C almost always fails by having its output stuck-at-0 (because it is designed differently to drive an external load). Given that knowledge, it is unlikely that inverter A is failing, as its most common fault does not explain the symptom. If operating correctly, A 's output should be one, so A being stuck-at-1 would not explain the incorrect value being observed at the device's output. However, the likely failures of inverters B and C are consistent with the symptom since either explains the deviation from expected behavior. Hence the diagnostician should measure at Y next to determine which of the two inverters is failing.

The objective of the the diagnosis dictates the granularity of Sherlock's analysis. Sometimes the objective is to identify which components are failing and how. Sometimes the task is simply to identify the failing components so that they can be replaced. Sometimes the task is to identify all the behavioral modes (good and bad). Sometimes diagnosis considers multiple test vectors, and sometimes there is only one. To accommodate these possibilities Sherlock must be told which modes it must discriminate among.

It is important to note that even if it is diagnostically unimportant to distinguish between some behavioral modes, knowledge of behavioral modes still helps Sherlock. Suppose a component has two faulty modes, M_1 of high probability, and M_2 of low probability, which we are not interested in discriminating between. If a measurement eliminates M_1 from consideration then the (*posterior*) probability that the component is faulty becomes low.

4 Framework

This section presents the overall framework including definitions for basic terminology and equations for computing the relevant probabilistic information. Section 5 presents heuristics for avoiding the combinatorial explosion resulting from moving from GDE to Sherlock.

The key conceptual extension to GDE is the introduction of behavioral modes. The extension is very easy as GDE can be viewed as having two behavioral modes (the good one and the faulty one with unspecified behavior) per component. In Sherlock there are simply more behavioral modes per component.

The *structure* of the device to be diagnosed specifies the components and their interconnections. Components are described as being in one of a set of distinct *modes*, where each mode captures a physical manifestation of the component (e.g., a valve being open, closed, clogged or leaky). The behavior of each component is characterized by describing its behavior in each of its distinct modes. We require that a component can be in only one mode at a time. We also require that a faulty component must remain in the same mode for all test vectors (in the exceptional case where a fault cannot be modeled this way, its behavior is captured by the unknown mode). Other than these there are very few restrictions on behavior models: a model can make incomplete predictions, the set of modes can be incomplete, and the predictions of different modes can overlap.

A component's modes consists of a set describing the component's proper behavior (e.g., the valve being on or off), and a set describing faulty behavior (e.g., the valve being clogged or leaky). When there is only one mode for proper behavior we abbreviate it as G (for "good"). Formally, a behavioral mode is a predicate on components, which is true of a device exactly when the device is in that behavioral mode. Every component has an unknown mode, U with no behavioral model, representing all (failure) modes whose behaviors are unknown.

In our example, we consider four behavioral modes of a digital inverter: good (abbreviated G), output stuck-at-1 (abbreviated $S1$), output stuck-at-0 (abbreviated $S0$), and an unknown failure mode (abbreviated U). The axioms for the behavior of the inverter are:

$$\begin{aligned} \text{INVERTER}(x) \rightarrow [& \\ & [G(x) \rightarrow [IN(x) = 0 \equiv OUT(x) = 1]] \wedge \\ & [S1(x) \rightarrow OUT(x) = 1] \wedge \\ & [S0(x) \rightarrow OUT(x) = 0]] \end{aligned}$$

The unknown behavioral mode $U(x)$ has no model.

Given the model library and the device structure Sherlock directly constructs a set of axioms SD , called the *system description* [13].

An *observation* is a set of literals describing the outcomes of measurements (e.g., $\{I = 0, X = 1, O = 0\}$) for a test vector which has been applied to the device. The *evidence* consists of a set of observations (e.g., $\{\{I = 0, X = 1, O = 0\}, \{I = 1, O = 0\}\}$).² This definition of allows us to incorporate accumulated evidence from different test vectors.

A *candidate* assigns a behavioral mode to every component of the device. Intuitively, a diagnosis is a candidate that is consistent with the evidence, however, we distinguish between a diagnosis for a particular observation and a diagnosis for all the evidence. A *diagnosis for an observation* is a candidate that is consistent with the observation — formally, that the union of the system description, the candidate, and the observation is logically consistent³. Formally a candidate is a set of literals, e.g., $\{G(A), G(B), U(C)\}$. To distinguish sets representing candidates we write $[G(A), G(B), U(C)]$. Note that in GDE a candidate is represented by the set of failing components, while in Sherlock a candidate is represented by a set that assigns a behavioral mode to every component. Thus, the Sherlock candidate $[G(A), G(B), U(C)]$ corresponds to the GDE candidate $[C]$.

In combining information from different observations we need to treat good and bad modes differently. By definition, a component manifests the same failure mode throughout all observations. However, if a component is in a good mode (e.g., valve is on) in one observation there is no reason to believe it should be in the same good mode for

²The process of generating good test vectors is outside the present scope of our theory.

³Note that by this definition some candidates may be eliminated as diagnoses on the basis of no observations whatsoever. For example, consider hypothetical models for two inverters in series where the first inverter had a mode output-stuck-at-1 and the second had a mode input-stuck-at-0. Note also that the candidate in which every component is operating in its unknown mode is always a candidate unless the combination of the system description and any observation by itself is inconsistent.

another test vector. If components have only a single good mode, combining information from multiple test vectors is straight-forward. Namely, a *diagnosis for the evidence* is a set of literals such that for every observation, the union of the system description, the candidate, and the observation is logically consistent. For brevity we operate within one observation, in the remainder of this paper, unless otherwise indicated. However, it is important to bear in mind that many of the design decisions underlying Sherlock only make sense when multiple observations are taken into consideration.

Like GDE, we make the basic assumption that components fail independently (which is sometimes unfounded) and that the prior probabilities of finding a component in a particular mode are provided. Recall that, although the behaviors of the different modes may sometimes overlap, we require that each mode captures a distinct physical state or condition of the component. Thus, the probabilities of all the modes of a component always sum to one. Under these assumptions, the prior probability that a particular candidate C_i is the actual one is:

$$p(C_i) = \prod_{m \in C_i} p(m).$$

where $p(m)$ denotes the prior probability of behavior mode m being manifested (i.e., a particular component being in a particular mode).

As candidates are eliminated, the probabilities of the remaining diagnoses must increase. (On occasion a candidate is eliminated purely as a result of the device's topology in which case the probability is adjusted by a renormalization.) Usually candidates are eliminated as a result of measurements. Bayes rule allows us to calculate the conditional probability of the candidates given that point x_i is measured to be v_{ik} (unless otherwise indicated, all probabilities are conditional on evidence previously accumulated. See [6] for more details):

$$p(C_i | x_i = v_{ik}) = \frac{p(x_i = v_{ik} | C_i) p(C_i)}{p(x_i = v_{ik})}$$

The denominator, $p(x_i = v_{ik})$, is just a normalization. $p(C_i)$ was computed as a result of the previous measurement (or is the prior). Finally, $p(x_i = v_{ik} | C_i)$ is determined as follows:

1. If $x_i = v_{ik}$ is predicted by C_i given the evidence so far then $p(x_i = v_{ik} | C_i) = 1$.
2. If $x_i = v_{ik}$ is inconsistent with C_i and the evidence then $p(x_i = v_{ik} | C_i) = 0$.
3. If $x_i = v_{ik}$ is neither predicted by nor inconsistent with C_i and the evidence then we make the presupposition (sometimes invalid) that every possible value for x_i is equally likely. Hence, $p(x_i = v_{ik} | C_i) = \frac{1}{m}$ where m is the number of possible values x_i might have (in a conventional digital circuit $m = 2$). Intuitively, this provides a bias for candidates which predict a measurement over those that don't.

Throughout the diagnostic session, the probability of any particular observation $x_i = v_{ik}$ is bounded below by the sum of the current probabilities of the candidates that entail it and bounded above by one minus the sum of the

current probabilities of the candidates that are inconsistent with it. See [6] for the estimate used. Similarly, the probability that a component is in a particular mode is given by the sum of the current probabilities of the candidates in which it appears.

4.1 Varieties of diagnostic tasks

In order to determine what next measurement is likely to provide the most information, Sherlock must determine the likelihood of hypothetical measurement outcomes and its consequences on the candidate space. The different diagnostic objectives dictate differing scoring functions. Sherlock is asked to discriminate among some modes and not others; by supplying Sherlock with sets of discrimination specifications — (a set of modes that are not to be discriminated). The discrimination specification partitions the diagnoses into a set of d -partitions. The goal of diagnosis is to identify the probable d -partitions and to suggest measurements which best pinpoint the actual one. For example, it may only be important to discriminate between good and faulty behavior. In this case, the most probable d -partition identifies which components have to be replaced. In the simple case where the objective is to discriminate among all behavioral modes, then every d -partition is just a singleton set consisting of a single diagnosis. Note that, in general, the different diagnoses within a single d -partition make different predictions. Although it may be unimportant to discriminate among them as far as the overall diagnostic objective is concerned, it is important to keep them separate to correctly compute the probabilities of measurement outcomes.

The specific approach used to select measurements is a *minimum entropy* technique — pick that measurement to make next that will yield, on average, the minimum entropy H (or conversely that measurement which extracts maximum information):

$$H = - \sum p(D_i) \log p(D_i).$$

Where $p(D_i)$ is the probability of a d -partition given evidence. This, in turn, requires computing the candidate probabilities given a hypothetical outcome. Fortunately this is computable fairly directly using Bayes rule (see [6] for details). The expected entropy resulting from measuring x_i is:

$$H_e(x_i) = \sum_{k=1}^{k=m} p(x_i = v_{ik}) H(x_i = v_{ik}),$$

where v_{ik} are the possible measurement outcomes and $H(x_i = v_{ik})$ is the entropy of the resulting set of d -partitions. Information theory tells us that, given certain assumptions, the measurement chosen by this scoring function will on average enable Sherlock to make the fewest number of measurements to identify the actual d -partition to a certain level of confidence. This approach (see examples in [6]) almost always suggests the optimum measurement common sense would suggest. The subsequent examples restate entropy as a cost function: ideal measurements have 0 cost, and useless measurements have cost 1.

If there are multiple test vectors, far greater care must be taken. Suppose the objective is to identify the faulty

components and how they are faulted. In this case Sherlock need only discriminate among faulty modes. The d -partitions for the overall objective are the intersection of those obtained from each of the multiple test vectors. In computing $H_e(x_i)$ we must take care to use these global d -partition, but only use the relevant candidates for determining $p(x = v_{ik})$ for a test vector. Thus, Sherlock identifies not only the best place to measure but also the best test vector (given the test vector set with which it has been supplied) under which to make the measurement.

4.2 Algorithms common to GDE and Sherlock

Sherlock, like GDE, exploits an assumption-based truth maintenance system (ATMS)[5]. Every literal stating that some component is in some behavioral mode is represented by an ATMS assumption. A literal indicating measurement outcome (e.g., $IN(A) = 0$) is represented by an ATMS premise⁴. The underlying Sherlock algorithms are similar to those of GDE except components can have multiple modes.

Sherlock computes the diagnoses by first constructing a set of conflicts. A *conflict* is a set of component behavioral modes which is inconsistent with the system description and some observation (i.e., a conflict is represented by an ATMS nogood). A conflict contains at most one behavioral mode per component. As in GDE, we represent the set of conflicts compactly in terms of the minimal conflicts, since conflicts are ordered by set-inclusion: every superset of a conflict is necessarily a conflict as well.

Intuitively, a minimal conflict identifies a small kernel set of component behavioral modes which violates some observation. It is easily shown that a candidate is a diagnosis iff it does not contain any minimal conflict. Thus, the complete set of diagnoses is computable from the minimal conflicts alone. Thus, Sherlock attempts to determine the minimal conflicts (in ATMS terminology these are minimal nogoods) as these provide the maximum diagnostic information.

Sherlock is typically used with a sound but incomplete prediction facility. Although soundness guarantees the conflicts Sherlock discovers are indeed conflicts, incompleteness sometimes makes it impossible to identify the minimal conflicts and consequently fails to rule out candidates as diagnoses. In the rest of this paper by minimal conflicts we simply mean the set of unsubsumed conflicts found by Sherlock, and by diagnosis we mean a candidate not ruled out by one of these conflicts. The consequences of incompleteness are not catastrophic and usually result in only a minimal degradation in diagnostic performance. This issue is discussed in more detail in [6].

In order to select the next measurement (and under which test vector) to make, Sherlock must evaluate the effects of a hypothetical measurement. To do so, Sherlock must be able to determine what possible outcomes hold in which candidates. Sherlock computes the sets of behavioral modes which support each possible outcome. If an outcome follows from a set of behavior modes, then it necessarily follows from any superset. Therefore, Sherlock need only record with each possible outcome the minimal

⁴To implement the search strategy discussed in the next section these literals have to be assumptions as well but this is outside the scope of this paper.

sets of behavior modes upon which it depends. Thus a possible measurement outcome holds in a candidate if a set of behavioral modes supporting the outcome is a subset of the candidate. Each set of behavioral modes supporting an outcome is represented by an ATMS *environment* and the set of all environments for an outcome is represented by an ATMS *label*. The details for this algorithm can be found in [5, 6]. In a later section we work through a simple example illustrating Sherlock's functioning.

5 Controlling the combinatorics

The presence of behavioral modes has two immediate consequences affecting the algorithms: (1) there are far more behavioral modes to reason about, and (2) the concept of minimal diagnoses which was so useful to GDE is now virtually meaningless. For example, if there are n components, each with k behavioral modes, there are k^n candidates which might have to be considered (as opposed to GDE's 2^n). Together these consequences make Sherlock significantly slower than GDE. This potential combinatorial explosion manifests itself in two ways in Sherlock. First, the set of conflicts, as well as the sets of behavioral modes underlying possible outcomes (i.e., the ATMS labels, explodes). This causes the prediction phase of Sherlock to explode. Second, the number of possible diagnoses is exponential, causing candidate generation to explode. Thus, the Sherlock architecture adds two tactics beyond those used in GDE to keep the combinatorial explosion under control. These tactics apply to GDE as well as Sherlock.

The focussing tactics do not affect the set of diagnoses produced (or probabilities ratios among them). We first present our strategy for the diagnostic objective of identifying all fault modes, and then later show it can be modified to find the best d -partitions. The basic idea is to focus reasoning to the subset of the diagnoses (called *leading diagnoses*) that satisfy the following conditions:

- All leading diagnoses have higher probability than all non-leading diagnoses.
- There are no more than k_1 (usually $k_1 = 5$) leading diagnoses. The exception is that all diagnoses having probability approximately equal to the k_1 th diagnosis are included (to accommodate roundoff difficulties).
- Candidates with probability less than $\frac{1}{k_2}$ th (usually $k_2 = 100$) of the best diagnosis are not considered.
- The diagnoses need not include more than k_3 (usually $k_3 = .75$) of the total probability mass of the candidates.

This approach focusses candidate generation to a small tractable set of leading diagnoses.

The primary remaining source of combinatorial explosion is the size of the ATMS labels for Sherlock's predictions. This is dealt with using a generalization of the focussing strategies outlined in [8] and are similar to some suggested in [7]. To handle this both the ATMS and the underlying constraint propagator used by Sherlock are restricted to focus their reasoning only on the leading diagnoses or tentative leading diagnoses. No prediction is made unless its results hold (i.e., one of its environments

is a subset of some focus environment) in the current focus. Furthermore, no environment is added to any ATMS label unless it holds in some current focus. If the ATMS discovers an environment not part of any current diagnosis, it does not add it to the prediction's label and instead stores it on its "blocked" label.

Unfortunately, there is a bootstrapping problem. The leading diagnoses cannot be accurately identified without sufficient minimal conflicts. The reasoning cannot produce enough minimal conflicts unless there are leading diagnoses to focus on. Another complication is that Sherlock cannot correctly evaluate the probability of a candidate via Bayes rule unless it is in the focus.

The following is an outline of the procedure Sherlock uses to identify the leading diagnoses and consists of a backtracking best-first search coupled with focussing tactics just discussed. The normalization factor of Bayes rule is left out in the search since it does not change the probability ordering of diagnoses and is the same for all candidates. The search estimates the probability of a tentative diagnosis — a candidate which is consistent with the predictions (more precisely contains no known conflict as a subset), but which has not yet been focussed upon — to be simply its prior probability (corrected by the normalization). This is an upper bound of its correct probability. Focussing the attention of the predictor on the tentative diagnosis might produce a conflict which eliminates it (i.e., drives its probability to zero) or it might be discovered that the diagnosis does not predict every measurement outcome (in which case its probability needs to be adjusted downwards by Bayes rule). Using these techniques the following search guarantees that it finds the same leading diagnoses an unfocussed Sherlock would find.

1. If, according to the criteria, there are sufficient leading candidates, stop. Let b be the upper-bound of the probabilities of the diagnoses which are reachable from the next place to push the best-first search forward. The key test is: is b less than the leading candidates?
2. Continue a best-first search for the next highest-probability (estimated by its upper bound) candidate which accounts for all the minimal conflicts.
3. Focus the predictor on the candidate (i.e., by unblocking the ATMS labels and permitting consumer execution). This finds any conflicts. It also finds any new predictions which follow from this candidate but which haven't been discovered earlier.
4. If the candidate contains a conflict, go to step 1.
5. Compute the probability of the candidate according to Bayes rule by multiplying its probability by $\frac{1}{m^n}$ where n is the number of times the candidate fails to predict some measurement outcome.
6. Go to 1.

This search may find more than the required number of diagnoses because the corrected probability of a best next candidate may be much lower than estimated. Although such candidates are diagnoses, they are not necessarily the leading ones.

Thus far we presumed that it is important to discriminate between all modes and that the d -partitions are the simple diagnoses. If it is not important to discriminate

among certain modes, the preceding algorithms must be modified to identify d -partitions.

To identify d -partitions efficiently requires some subtle changes to the best-first search. Whenever a diagnosis is found, all the other candidates potentially in the same d -partition must be identified to fill out the d -partition and accurately calculate its probability. However, this alone is insufficient to ensure that our previous algorithm finds the best d -partitions because the probability of the best diagnoses within a d -partitions is only a lower bound on the probability of the d -partition it could be part of. Therefore, we must modify the search such that each diagnoses is scored (only for the purposes of the search) by an upper bound of the probability of the d -partition it is part of. For each component, the 'probability' score assigned to each mode is the sum of the prior probabilities of that mode and all modes later in the mode ordering among which Sherlock is not required to discriminate. As a result of this ordering the 'probability' of a diagnosis is an upper-bound of the d -partition of which it is a member. Once one diagnosis of a d -partition is found, the remaining members of the d -partitions are filled out and its probability correctly computed. (Sherlock incorporates heuristics that avoid filling out the d -partition with extremely low probability diagnoses.) As a result, Sherlock finds the leading d -partitions meeting the criteria for simple diagnoses just laid out.

6 A simple example

To demonstrate the basic ideas of Sherlock's operation with fault modes consider the three inverter example. We set the focussing thresholds as indicated earlier and the diagnostic objective to identify the mode of every component. However, as this example is tractable without using any focussing heuristics, we also show the correct values (i.e., having computed all the diagnoses) in parentheses. Suppose every inverter is modeled as described earlier, with A and B tending to fail stuck-at-1 and C tending to fail stuck-at-0:

A	B	C
$p(G(A)) = .99$	$p(G(B)) = .99$	$p(G(C)) = .99$
$p(S1(A)) = .008$	$p(S1(B)) = .008$	$p(S1(C)) = .001$
$p(S0(A)) = .001$	$p(S0(B)) = .001$	$p(S0(C)) = .008$
$p(U(A)) = .001$	$p(U(B)) = .001$	$p(U(C)) = .001$

With no observations Sherlock finds the single leading diagnosis (all diagnosis probabilities are normalized):

$$p[G(A), G(B), G(C)] = 1(.970).$$

The unfocussed Sherlock finds $4^3 = 64$ diagnoses (as there are no symptoms anything could be happening). Given a zero input ($I = 0$), Sherlock computes the following outcomes and their supporting environments. The final column indicates the additional environments an unfocussed Sherlock discovers):

$X = 0,$	$\{S0(A)\}$
$X = 1,$	$\{G(A)\} \{S1(A)\}$
$Y = 0,$	$\{G(A), G(B)\} \{S1(A), G(B)\} \{S0(B)\}$
$Y = 1,$	$\{S0(A), G(B)\} \{S1(B)\}$
$O = 0,$	$\{S0(A), G(B), G(C)\} \{S1(B)G(C)\} \{S0(C)\}$
$O = 1,$	$\{G(A), G(B), G(C)\} \{S1(A), G(B), G(C)\} \{S0(B), G(C)\} \{S1(C)\}$

The first line states that $X = 0$ under assumption $S0(A)$, or equivalently that $X = 0$ in every candidate which includes $S0(A)$. However, the focussed Sherlock finds no label for $X = 0$ as it does not hold in the single leading diagnosis. Intuitively, the last line states that the output is one if either (1) all the components are good (which is the leading diagnosis), (2) the first inverter is stuck-at-1 and the other two are good, (3) the second inverter is stuck-at-0, and the final inverter is good, or (4) the last inverter is stuck-at-1.

If we apply the minimum entropy technique we find costs ($\$$ denotes the cost function, and the focussed Sherlock cost is shown first followed by the correct cost in parentheses):

$$\$(X) = 1(.99), \$(Y) = 1(.95), \$(O) = 1(.91).$$

All these costs are high because there is no evidence that a fault necessarily exists. The costs are all equal for the focussed Sherlock because there is only one leading diagnosis and therefore nothing can be learned by further measurement.

Suppose O is measured to be zero. There are four minimal conflicts (because each set of the minimal environments supporting $O = 1$ is now a conflict):

$$\{G(A), G(B), G(C)\} \{S1(A), G(B), G(C)\} \\ \{S0(B), G(C)\} \{S1(C)\}$$

Sherlock notices the leading candidate is eliminated and continues best-first search to find the following leading diagnoses:

$$p([G(A), G(B), S0(C)]) = 0.432(.426) \\ p([G(A), S1(B), G(C)]) = 0.432(.426) \\ p([S0(A), G(B), G(C)]) = 0.054(.053) \\ p([G(A), G(B), U(C)]) = 0.027(.027) \\ p([U(A), G(B), G(C)]) = 0.027(.027) \\ p([G(A), G(C), U(B)]) = 0.027(.027) \\ \text{Total} = .999(.986)$$

The next highest probability diagnoses being:

$$p([S1(A), G(B), S0(C)]) = .003(.003).$$

The figures in parentheses indicate the correct probabilities (initially there were 64 diagnoses, now 42 diagnoses remain) and Sherlock has identified the leading ones.

There are four important things to notice about this list of leading diagnoses. First, even though the Sherlock algorithm is running with $k_1 = 5$, it finds 6 candidates because it expands its search slightly to make sure candidates are not eliminated simply by round off errors. Second, the top 6 of the 42 candidates contain 98.6% of the probability mass. Third, the heuristic estimates are quite accurate — they are equal to the correct values normalized by .986. Fourth, although diagnosis $[S0(A), G(B), G(C)]$ has the same prior probability of the three candidates $[G(A), G(B), U(C)]$, $[U(A), G(B), G(C)]$ and $[G(A), G(C), U(B)]$, after the two measurements its probability is twice that of the others. This is because the three candidates predict no value of the output and hence their posterior probability is reduced by one-half by Bayes rule.

Given the leading diagnoses, the resultant probabilities of behavior modes are:

	A	B	C
$p(G)$.918(.911)	.541(.538)	.541(.538)
$p(S1)$	0(.007)	.432(.434)	0
$p(S0)$.054(.054)	0(.0005)	.432(.435)
$p(U)$.027(.028)	.027(.027)	.027(.027)

The table indicates that the major failure modes to consider are C stuck-at-0, and B stuck-at-1 and that all other faults are unlikely.

The resulting ATMS labels are (for this simple example focussing no longer has any affect on labels):

$$\begin{aligned}
 X = 0, & \quad \{G(B), G(C)\} \{S0(A)\} \\
 X = 1, & \quad \{G(A)\} \{S1(A)\} \\
 Y = 0, & \quad \{G(A), G(B)\} \{G(B), S1(A)\} \{S0(B)\} \\
 Y = 1, & \quad \{G(C)\} \{G(B), S0(A)\} \{S1(B)\} \\
 & \quad \{S1(A), G(B), G(C)\} \{S0(B), G(C)\} \{S1(C)\}
 \end{aligned}$$

Suppose that we applied a second test vector with $I = 1$ (the first test vector was $I = 0$), and evaluated the hypothetical measurements:

$$\$(X_1) = .72(.72), \$(X_2) = .94(.92), \$(Y_1) = .31(.31)$$

$$\$(Y_2) = .91(.90), \$(O_2) = .89(.89).$$

Thus we see that measuring Y using the first test vector ($I = 0$) is the best measurement. This is because measuring Y will differentiate between the two high probability candidates. However, measuring O under the second test vector ($I = 1$) is useful as well. Suppose $O = 0$ under the second test vector. The resulting probabilities are:

$$p([G(A), G(B), S0(C)]) = 0.450(.444)$$

$$p([G(A), S1(B), G(C)]) = 0.450(.444)$$

$$p([S0(A), G(B), G(C)]) = 0.056(.056)$$

$$p([G(A), G(B), U(C)]) = 0.014(.014)$$

$$p([U(A), G(B), G(C)]) = 0.014(.014)$$

$$p([G(A), G(C), U(B)]) = 0.014(.014)$$

Although measuring $O = 0$ again does not eliminate any diagnosis, it provides further evidence that a component is not behaving in some unknown mode, thus slightly raising the probabilities of the first three diagnoses.

7 Acknowledgments

We appreciate discussion and comments from Daniel G. Bobrow, Brian Falkenhainer, Adam Farquhar, Alan Mackworth, Sanjay Mittal, Olivier Raiman, Mark Shirley and Peter Struss on the topics of this paper.

References

- [1] Brown, J.S., Burton, R. R. and de Kleer, J., Pedagogical, natural language and knowledge engineering techniques in SOPHIE I, II and III, in: D. Sleeman and J.S. Brown (Eds.), *Intelligent Tutoring Systems*, (Academic Press, New York, 1982) 227-282. An expansion of the relevant sections of this paper appears in these readings under the title, Model-based diagnosis in SOPHIE III.
- [2] Davis, R., Diagnostic Reasoning based on structure and behavior, *Artificial Intelligence* 24 (1984) 347-410. Appears in these readings.
- [3] Davis, R., and Hamscher, W., Model-based reasoning: Troubleshooting, in *Exploring artificial intelligence*, edited by H.E. Shrobe and the American Association for Artificial Intelligence, (Morgan Kaufman, 1988), 297-346. Appears in these readings.
- [4] de Kleer, J., Local methods of localizing faults in electronic circuits, Artificial Intelligence Laboratory, AIM-394, Cambridge: M.I.T., 1976. An expansion of the relevant sections of this paper appears in these readings under the title, Model-based diagnosis in SOPHIE III.
- [5] de Kleer, J., An assumption-based truth maintenance system, *Artificial Intelligence* 28 (1986) 127-162. Also in *Readings in NonMonotonic Reasoning*, edited by Matthew L. Ginsberg, (Morgan Kaufman, 1987), 280-297.
- [6] de Kleer, J. and Williams, B.C., Diagnosing multiple faults, *Artificial Intelligence* 32 (1987) 97-130. Also in *Readings in NonMonotonic Reasoning*, edited by Matthew L. Ginsberg, (Morgan Kaufman, 1987), 372-388. Appears in these readings.
- [7] Dressler, O. and A. Farquhar, Problem solver control over the ATMS, Siemens Technical report INF 2 ARM-13-89, Munich, 1989.
- [8] Forbus, K.D. and de Kleer, J., Focusing the ATMS, *Proceedings of the National Conference on Artificial Intelligence*, Saint Paul, MN (August 1988), 193-198.
- [9] Genesereth, M.R., The use of design descriptions in automated diagnosis, *Artificial Intelligence* 24 (1984) 411-436. Appears in these readings.
- [10] Hamscher, W.C., Model-based troubleshooting of digital systems, Artificial Intelligence Laboratory, TR-1074, Cambridge: M.I.T., 1988.
- [11] Holtzblatt, L.J., Diagnosing multiple failures using knowledge of component states, *IEEE Proceedings on AI applications*, 1988, 139-143. Appears in these readings.
- [12] Raiman, O., Diagnosis as a trial: The alibi principle, IBM Scientific Center, 1989. Appears in these readings.
- [13] Reiter, Raymond, A theory of diagnosis from first principles, *Artificial Intelligence* 32 (1987) 57-95. Also in *Readings in NonMonotonic Reasoning*, edited by Matthew L. Ginsberg, (Morgan Kaufman, 1987), 352-371. Appears in these readings.
- [14] Struss, P., A framework for model-based diagnosis, Siemens TR INF2 ARM-10-88. Appears in these readings.
- [15] Struss, P., and Dressler, O., "Physical negation" — Integrating fault models into the general diagnostic engine, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI (August 1989). Appears in these readings.