01 Jun 1988

Automated Circuit Diagnosis using First Order Logic Tools

Barbara Smith

Ralph W. Wilkerson
Missouri University of Science and Technology, ralphw@mst.edu

Gerald E. Peterson

McDonnell Douglas

Recommended Citation

B. Smith et al., "Automated Circuit Diagnosis using First Order Logic Tools," *Proceedings of the 1st International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 1988*, vol. 1, pp. 456 - 463, Association for Computing Machinery, Jun 1988.
The definitive version is available at https://doi.org/10.1145/51909.51961

Automated Circuit Diagnosis using First Order Logic Tools

Barbara Smith and Ralph Wilkerson
Department of Computer Science
University of Missouri-Rolla
Rolla, MO 65401

Gerald E Peterson
McDonnell Douglas
Aerospace Information Services Company
W401/105/2/A1
P.O. Box 516
St. Louis, MO 63166

Abstract

While numerous diagnostic expert systems have been successfully developed in recent years, they are almost uniformly based on heuristic reasoning techniques (i.e., shallow knowledge) in the form of rules. This paper reports on an automated circuit diagnostic tool based on Reiter's theory of diagnosis. In particular, this is a theory of diagnosis based on deep knowledge (i.e., knowledge based on certain design information) and using first order logic as the representation language. The inference mechanism which is incorporated as part of the diagnostic tool is a refutation based theorem prover using rewriting systems for Boolean algebra developed by Hsiang. Consequently, the diagnostic reasoning tool is broadly based on Reiter's model, but incorporates complete sets of reductions for Boolean algebra to reason over equational descriptions of the circuits to be analyzed. The refutational theorem prover uses an associative commutative identity unification algorithm described by Hsiang, but requires additional focusing techniques in order to be appropriate for diagnosing circuits. A prototype version of the mainline diagnostic program has been developed, and has been successfully demonstrated on several small but nontrivial combinational circuit examples.

1. Introduction

Significant use is now made of the computer as a tool to aid in the design and manufacture of complex devices. When these devices fail, the complexity makes it difficult to diagnose the problem and determine the cause. It seems reasonable to make use of the tool which aided in the development of the design in the diagnostic process.

There are three parts to the diagnostic problem. The first is to determine whether a system is exhibiting the correct behavior in a given situation. The second aspect of diagnosis is to determine what could be causing the observed misbehavior. The third is that of diagnostic testing. When there is more than one potential diagnosis, it is the goal of diagnostic testing to establish tests which will confirm or eliminate some of the multiple diagnoses. The test designer may suggest measurements to be taken under current conditions or observations to be made under different conditions. For example, the output(s) of a circuit may be observed under a different set of inputs.

A change may be on the horizon for automated diagnostic reasoning. The first, and to this time most successful, attempts to replicate human performance in the area of diagnostic reasoning have involved using human experience and knowledge of the problem domain. Often, the human experience and knowledge is captured in the form of rules and is implemented as a rule-based expert system.

A different approach to diagnosis has been termed reasoning from first principles or reasoning from deep knowledge. In this approach, the automated diagnostician uses a description of the system structure and observations describing its performance to determine if any faults are apparent. If there is evidence that the system is faulty, the diagnostician uses the system description and observations to ascertain which component(s), if faulty, would explain the behavior. The first principles approach clearly address the first two aspects of the diagnostic problem. However, there are few results which address diagnostic testing within this framework.

Diagnosis from first principles is a more recent approach to diagnostic reasoning than the more well-known expert system approach. There are many expert systems in use and much is known about when to apply and how to construct expert systems. However, issues such as these as well as several others are open research areas in diagnosis from first principles. This work addresses some of these issues within the framework of diagnostic theory developed by Reiter [*Re*87].

Within the artificial intelligence community, a dichotomous relationship seems to exist between those supporting the use of diagnosis from first principles and those supporting the use of rule-based expert systems and other so-called shallow reasoning systems. It may well be that a hybrid approach to diagnosis ultimately will be the most successful. The strengths of one approach can cover some of the weaknesses of the other. Human experts in some domains, for example medicine, appear to use both methods in some cases. They first apply heuristics and experiential knowledge (a rule based approach) and when necessary, use a more precise model and reason from that model (a first principles approach) to determine the diagnosis. The RENAL system

This research was supported in part under grants from McDonnell Douglas Research Laboratories (Independent Research and Development) and the Missouri Research Assistance Act.

[*Ku*87] which is being developed to carry out medical diagnosis of kidney diseases combines aspects of the two approaches.

What is now termed reasoning from first principles first appeared in the work of de Kleer [*dK*76] on the diagnosis of faults in circuits. Major contributions to the development of this type of diagnostic reasoning have been made by Davis [*Da*84], de Kleer and Williams [*DW*87], Genesereth [*Ge*84], and Reiter [*Re*87]. The significance of diagnosis from first principles and more general reasoning about physical systems was recognized with the appearance of a special issue of the Journal of Artificial Intelligence (1984) devoted to reasoning about physical systems.

2. Diagnosis from first principles

The hallmark of diagnosis from first principles is the use of information about the structure of the device, its component parts and interconnections, and a description of its intended behavior. The fact that the system description consists exclusively of information about correct behavior is significant. Thus, it is not necessary to know the ways in which a device or component might fail in order to use the approach of diagnosis from first principles. It is assumed that the design has been verified. The diagnostic process is not intended to uncover design flaws, but certainly the general scrutiny of the process may lead to the discovery of such flaws by the human diagnostician.

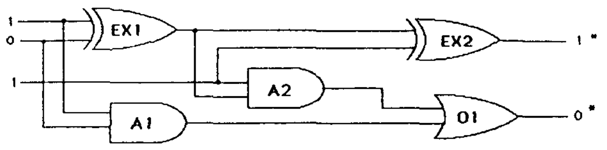
It is also assumed that the device is manufactured correctly and that the physical structure matches the design description. In general, automated diagnosticians reasoning from design information cannot identify problems such as bridge faults where there are connections present in the device which are not represented in the design description. Davis [*Da*84] presents a method for diagnosing this type of fault.

As will be seen, the system description of a device is quite detailed and the description of even small devices is very large. However, there are two aspects of the design process which can lessen these problems.

First, complex devices are often designed in a hierarchical manner. The general, overall structure is first developed illustrating the interconnections of large components. The design of these components is then treated as a separate design problem. Diagnosis can be handled in the same hierarchical fashion. The fault is first isolated to a large component with the diagnosis of the fault(s) within that component handled as a different diagnostic problem. Genesereth [*Ge*84] proposed that diagnosis be a hierarchical process. However, the complexity of the diagnostic problem is not uniform across all levels of the hierarchy. High level components generally have complex functions and behavior which make it more difficult to isolate the fault. Nonetheless, the extremely large number of subcomponents at the lower levels in the hierarchy make the hierarchical approach effective.

Secondly, many complex devices, particularly electronic components, are designed with the use of computer-aided design tools. The results of interactive design sessions can form the basis of the design description information required by the automated diagnostician. Diagnosis is not tied to any single description format. Any description which captures the structure and operation of the device and can be used by the reasoning component of the diagnostician is acceptable. Thus, most hardware description languages are suitable for describing the device. The development of standardized hardware description languages should prove valuable to diagnosis from first principles.

In the following sections, diagnosis from first principles is presented as developed by Genesereth, de Kleer and Williams, and Reiter. The approach of each of these researchers is first presented individually and then comparisons are drawn. One device appears throughout the literature to illustrate diagnosis. This device is the full adder which is illustrated in Figure 1. Genesereth's work is described in more detail than the work of de Kleer and Williams. Much of what is described in conjunction with Genesereth's work is common to the work of the others and will be referred to in the summary of their work.



FULLADDER

* - indicates incorrect value

DIAGNOSIS ((EX1) (EX2 O1) (EX2 A2))

Figure 1

Diagnostic Method of Genesereth

The major reference for Genesereth's work is [*Ge*84] which is an expanded version of [*Ge*82]. Genesereth has developed an automated diagnostic program called DART. A "tester" who can observe and possibly manipulate the device to be diagnosed provides the diagnostic session information to DART. The significant contributions resulting from the development of DART are the use of a hierarchical approach to the diagnostic process and in the area of diagnostic testing. Genesereth proposes a method for suggesting tests, the results of which can be used to decrease the number of possible diagnoses.

The design description language and notation which is used by the DART program is based on prefix predicate calculus. Genesereth provides the description (structural and behavioral) for the full adder. The lowest level components of the device are the EXCLUSIVE-OR gates EX1 and EX2, the AND gates A1 and A2, the OR gate O1, and the connections between these components. The fact that all connections must be explicitly defined affects the type of faults which can be diagnosed.

A proposition defining connections to be ideal is included. Thus the program cannot diagnose faulty behavior resulting from non-ideal connections, for example, shorts. Theoretically, the DART program could reason with the possibility that connections other than those explicitly defined existed. However, this would necessitate the use of non-monotonic reasoning and result in a much larger number of possible diagnoses for the observed system behavior. Abstractions such as the assumption of ideal connections are necessary to keep the diagnostic problem tractable. However, as in any application where details are suppressed, there is the risk that the model does not truly reflect the structure and operation of the device.

Genesereth identifies four types of information present in the design description. *Theoretical information* is what has been described thus far: components, connections, functional behavior of components, etc.. *Achievable data* are conditions which the tester can control, for example, setting inputs to particular values. *Observable data* correspond to information which can be obtained by the tester but not controlled. An example of this type of data is the observed output(s) of a circuit. The fourth type of information is the collection of simplifying *assumptions* to be used in the diagnostic process.

As discussed above, assumptions are necessary to keep the diagnostic problem to a reasonable size. It may be that advances in the theory of diagnosis and the development of more powerful automated reasoning systems will lessen the need for these assumption. This is the case with the single fault assumption. More recently developed automated diagnosticians [*Re*87], [*DW*87] can operate effectively without this assumption. The assumptions which are commonly applied in a diagnostic setting are the single fault assumption, the non-intermittency assumption, and the assumption that certain parts of the device are not faulted. The statements that describe connections as ideal are examples of this last assumption.

The single fault assumption states that if a component is faulted, then only that component is faulted and all other components are functioning correctly. The non-intermittency assumption requires that devices and components behave consistently when presented with the same conditions at a different time. Genesereth provides formal characterizations of each of the assumptions which the user can include or exclude as appropriate for the domain of the diagnostic problem. However, the situation is more complex than Genesereth leads his readers to believe.

It is not simply a matter of modifying the device description by adding or deleting a few statements. Whether or not an assumption is included can determine the type of reasoning of which the diagnostician must be capable. While the assumptions do represent efficiency concerns, as Genesereth characterizes them, they also determine how powerful the reasoning component of the diagnostician must be and the computational complexity of the process.

The diagnostic operation of DART is based on the determination and explanation of symptoms. A symptom is observable data which is inconsistent with the system design and achievable data. A symptom might be, for example, observing an output value to be 1 when the value of the input(s) and the proper operation of the device predict it to be 0. Based on one or more symptoms, DART computes a suspect set within which the faulty component(s) lie.

Given a suspect set, DART suggests tests to be carried out to distinguish among the suspects. Genesereth defines tests to consist of "zero or more propositions to be achieved and at least one proposition to be observed". Thus a test will supply additional measurements or observed data under a different set of conditions. The difficulty of suggesting tests lies in determining a test which will provide new information and whose outcome depends on the suspected components.

Let $a_1, \dots, a_m$ be the achievable and *ob* the observable data prescribed by the test. A check which DART carries out to determine whether any new information may be provided by the test is to try to prove each of the following propositions.

(IF (AND $a_1, \dots, a_m$) *ob*)
(IF (AND $a_1, \dots, a_m$) (NOT *ob*))

If either can be proved, the test will provide no new information. Reiter proves the validity of this check within the context of his formal theory of diagnosis.

The above "novelty check" as Genesereth calls it is useful, but much stronger criteria need to be developed. The area of measurement and testing theory is virtually unexplored. Genesereth recognizes that considerations such as the cost of a test should be taken into account. Also needed is a means of estimating the value of a test based on its ability to decrease the number of possible diagnoses. Obviously, a test which is expensive to carry out and does little to prune the suspect set has little merit.

The DART system uses a method called resolution residue to carry out inferences. Reiter's theory of diagnosis establishes that the underlying inference mechanism can take any form so long as it is a sound decision procedure for the domain. A strength of the resolution residue procedure within DART is that its use provides a means of generating suggested tests.

Diagnostic Method of de Kleer and Williams

One of the major contributions of their work [*DW*87] is that multiple faults can be diagnosed. The other is a means of effectively decreasing the number of possible diagnoses. The work of de Kleer and Williams has a number of characteristics in common with the general theory of diagnosis developed by Reiter.

In the earlier diagnostic work, such as that done by Genesereth and Davis, the single fault assumption played a prominent role. Even if it was not strictly part of the model, it was necessary to keep the size of the diagnostic problem reasonable. The general diagnostic engine (GDE) of de Kleer and Williams can deal effectively with the complexity introduced by allowing multiple faults.

As in Genesereth's DART system, GDE is guided by symptoms, with a symptom implying which components might be faulty. These components form a *conflict* which is a set of components that "cannot all be functioning correctly". An obvious characteristic of a conflict is that a superset of it is also a conflict. However, GDE must find and manipulate minimal conflicts. The discovery of symptoms and minimal conflicts leads to the determination of what de Kleer and Williams call minimal candidates. These are diagnoses in Reiter's terminology.

The process of candidate generation is an incremental one of combining the information gained from new minimal conflicts with the known minimal candidates. Initially, the device is assumed to be working and the empty set is the first minimal candidate. The candidate space is viewed as a lattice. If a new minimal conflict is not explained by an existing candidate then that candidate is replaced by one or more of its supersets. The supersets consist of the candidate with one of the elements of the new conflict added. This candidate generation process divides the candidate space lattice into two parts. Above the dividing line are the supersets of the minimal candidates. Below the dividing line are the sets of components which do not explain the observations. The sets of components on the dividing line define the minimal candidates or diagnoses.

The above process allows for the diagnostician to identify multiple faults while at the same time guaranteeing that the multiple fault diagnoses found are minimal sets of components. Note that if the single fault assumption were to be applied, the task of candidate generation would be much simpler. All that would be necessary is that the intersection of the minimal conflicts be found. The members of the resulting set define the possible single faults.

A disadvantage of the candidate generation procedure is that it requires that minimal conflicts be determined by the inference mechanism. In order to guarantee minimality, the inference mechanism must explore potential conflicts in increasing order of size until an inconsistency is found. The inconsistency takes the form of a difference between a predicted and observed value or behavior. The inference mechanism of GDE contains a refinement which exploits the sparseness of the search space and makes minimal conflict discovery more efficient.

Diagnostic Method of Reiter

Raymond Reiter [*Re*87] has developed what he terms a theory of diagnosis. The goal of the theory development is to establish a firm foundation on which to develop automated diagnosticians. His theory is most general and is applicable to many areas of diagnosis. However, the focus of this paper is the diagnosis of circuit faults so examples and extensions to Reiter's work will be in that field.

The first point to note about Reiter's work is that it is theoretical in nature. He makes no statements to indicate that an automated diagnostician based on the theory has been built. De Kleer and Williams [*DW*87] also refer to Reiter's theory as unimplemented. Part of the work of this paper involves an implementation of a diagnostician based on Reiter's theory.

Because of the generality of Reiter's theory, issues which are the central focus of some of the earlier work on diagnosis from first principles are ignored. One of these issues is that of the representation logic. Since the theory is independent of the representation logic, the underlying theorem prover can be implemented in a manner appropriate for the diagnostic domain. In contrast, the diagnostic systems of Genesereth, Davis, and de Kleer and Williams seem to be dependent on a particular type of inference mechanism. However, Reiter has not demonstrated that the approach of his theory is more general than the approaches of these other researchers. The examples and representation which he uses are the same as those of the other researchers.

Some of the terminology of the diagnostic problem domain has already been presented. In the following section, this terminology is expanded and rephrased within the context of Reiter's general theory. The definitions are taken from [*Re*87]. We begin with the concept of a system which is to be diagnosed. This concept is central to the first principles approach. A *system* is a pair (SD, COMPONENTS) where SD is the system description represented as a set of first-order sentences and COMPONENTS is a finite set of constants representing the constituent parts of the system.

This approach to diagnosis uses the description of a correctly functioning set of components and does not assume any particular mode of failure. Thus the concept of a malfunctioning component must be very general. The predicate AB(component) is used for this purpose. Consider again the example of the full adder. A complete representation of the circuit which makes use of the ABnormal predicate is shown in Figure 2.

The name and type of each component is specified in components and gate types. The interconnections of the components are given, as are the observed input and output values. In the case of the full adder, all values are binary. This type of constraint might be absent or quite general, depending on the device to be diagnosed. For example, values might be constrained to be integer or positive in some application or without any constraint in another application. The correct behavior of each component as a function of its input(s) is described by the gate descriptions. In the case of the full adder, this is accomplished by the inclusion of function definitions for AND, OR, and EXCLUSIVE-OR gates.

The generality of the approach and representation does not preclude the use of domain specific information concerning faults. It is not necessary to know the ways in which a component can be faulted. However, Reiter states that if such information is available, it can be included in the system description. The general form of such information is:

$$\text{COMPONENT\_TYPE}(x) \wedge AB(x) \Rightarrow \text{FAULT}_1(x) \vee \ \dots \ \vee \text{FAULT}_n(x)$$

Also necessary for diagnosis is one or more sets of observations of the system. An *observation* is simply defined to be a finite set of first order sentences. In the full adder example, observations provide information about known input and output values, such as:

$$\begin{aligned} &\text{IN1(EX1)} = 1 \\ &\text{IN2(EX1)} = 0 \\ &\text{IN1(A2)} = 1 \\ &\text{OUT1(EX2)} = 1 \\ &\text{OUT1(O1)} = 0 \end{aligned}$$

Reiter does not address the problem of representing and reasoning about multiple sets of observations.

```
                    INPUT DATA FOR FULLADDER

(A1 A2 O1 EX1 EX2)            Components
((ANDG A1))               Gate Types
((ANDG A2))
((EXORG EX1))
((EXORG EX2))
((ORG O1))
((- ANDG x) (AB x) (= (OUT x) (AND (IN1 x) (IN2 x))))     Gate
((- EXORG x) (AB x) (= (OUT x) (EXOR (IN1 x) (IN2 x))))   Descriptions
((- ORG x) (AB x) (= (OUT x) (OR (IN1 x) (IN2 x))))
((- = 1 0))        Refutation Axioms
((- = 0 1))

                          DEMODULATORS

((IN1 EX2) (OUT EX1))
((IN2 A2) (OUT EX1))              Connections
((IN1 O1) (OUT A2))
((IN2 O1) (OUT A1))
((OUT O1) 0)                      Input/Output values
((OUT EX2) 1)
((IN1 EX1) 1)
((IN2 EX1) 0)
((IN1 A2) 1)
((IN1 A1) 1)
((IN2 A1) 0)
((IN2 EX2) 1)
((OR 1 x) 1)                      Logical operations
((OR x 1) 1)
((OR 0 0) 0)
((AND 0 x) 0)
((AND x 0) 0)
((AND 1 1) 1)
((EXOR x x) 0)
((EXOR 0 1) 1)
((EXOR 1 0) 1)
```

Figure 2

As discussed earlier, the goal of diagnostic work is to determine the component(s) which if ABnormal would explain the observed behavior. Since the system description and observations have an underlying logical representation, the concept of a diagnosis is tied to logical consistency. Formally, Reiter defines a *diagnosis* for a device with constituent COMPONENTS, and a system description SD under a set of observations OBS to be a minimal set $\Delta \subseteq$ COMPONENTS such that

$$\text{SD} \cup \text{OBS} \cup \{\neg \text{AB}(c) \mid c \in \text{COMPONENTS} - \Delta\} \cup \{\text{AB}(c) \mid c \in \Delta\}$$

is consistent. A slightly simpler characterization of a diagnosis for (SD, COMPONENTS, OBS) is a minimal set Δ such that $\text{SD} \cup \text{OBS} \cup \{\neg \text{AB}(c) \mid c \in \text{COMPONENTS} - \Delta\}$ is consistent. For a proof of the equivalence of the two definitions see [*Re*87].

Two major points arise from this definition. First, a diagnosis must be minimal. As will be seen, Reiter has developed an elegant means of identifying the minimal sets of components which form the diagnoses. Secondly, in order to identify a diagnosis, there must be a consistency test for the logic used in the representation. This second point presents a serious problem since, in general, there is no decision procedure for determining the consistency of a first order logic formula. Does Reiter's approach have any merit? The answer is yes. There is no decision procedure for the general question of consistency but for certain domains the question of consistency is decidable. This is true, for example, in the area of boolean circuits.

As will be shown, there are a number of similarities between the work of Reiter and that of de Kleer and Williams. For example, what Reiter terms a diagnosis, de Kleer and Williams refer to as a minimal candidate. The difference, however, between their work is not just a matter of nomenclature. Reiter's approach appears more general than that of de Kleer and Williams and provides a formal basis for studying diagnosis from first principles. In order to determine the diagnoses, Reiter makes use of the concept of a conflict set which was developed by de Kleer [*dK*76]. A *conflict set* for (SD, COMPONENTS, OBS) is a set $\{c_1, \ldots, c_k\}$ such that $\text{SD} \cup \text{OBS} \cup \{\neg \text{AB}(c_1), \ldots, \neg \text{AB}(c_k)\}$ is inconsistent. A conflict set is minimal if and only if no proper subset of it is a conflict set for (SD, COMPONENTS, OBS).

Reiter's procedure for determining diagnoses for (SD, COMPONENTS, OBS) is based on determining what he terms the minimal hitting sets for the collection of conflict sets for (SD, COMPONENTS, OBS). Define a minimal hitting set as follows:

> Let C be a collection of sets. A *hitting set* for C is set $H \subseteq \bigcup_{S \in C}$ such that $H \cap S \neq \{\}$ for each $S \in C$. A hitting set for C is minimal iff no proper subset of it is a hitting set for C.

The following theorem, which Reiter proves, ties together the concepts of minimal hitting sets, conflict sets and diagnoses.

> Theorem: $\Delta \subseteq$ COMPONENTS is a diagnosis for (SD, COMPONENTS, OBS) iff Δ is a minimal hitting set for the collection of conflict sets for (SD, COMPONENTS, OBS).

Thus the problem of computing diagnoses becomes one of computing the minimal hitting sets for the conflict sets of (SD, COMPONENTS, OBS). Note that the problem is phrased in terms of conflict sets not minimal conflict sets as is the case in the work of de Kleer and Williams. Since the conflict set returned by the reasoning component need not be minimal, the reasoning component may be simpler.

Reiter provides an elegant means of computing hitting sets through the use of a hitting set (HS) tree. Minimal hitting sets are determined by a pruned HS-tree. Reiter defines an HS-tree as follows:

> Let F be a collection of sets. An *HS-tree* for F is a smallest edge-labeled and node-labeled tree with the following properties:
>
> (1) The root is labeled by $\surd$ if F is empty. Otherwise the root is labeled by a set of F.
>
> (2) If n is a node of T, define $H(n)$ to be the set of edge labels on the path in T from the root node to n. If n is labeled by $\surd$ then it has no successor nodes in T. If n is labeled by a set Σ of F then for each $\sigma \in \Sigma$, n has a successor node n_σ joined to n by an edge labeled by σ. The label for n_σ is a set $S \in F$ such that $S \cap H(n_\sigma) = \{\}$ if such a set S exists. Otherwise, $\surd$ is the label for n_σ.

Reiter states that $H(n)$ for a node n labeled by $\surd$ is a hitting set for F and each minimal hitting set for F is $H(n)$ for some node n for which $\surd$ is the label. Since only minimal hitting sets are desired, a pruned HS-tree will be constructed in such a way that $H(n)$ for any node labeled by $\surd$ is a minimal hitting set.

Some concerns arise in applying the concept of an HS-tree to the process of computing diagnoses. First, the collection of sets F is not explicitly known. In the diagnostic process, F is the set of conflict sets for (SD, COMPONENTS, OBS). Secondly, the determination of a set $f \in F$ is computationally expensive since f is computed by the reasoning component. Thus it is necessary to use a method which incrementally builds and prunes the HS-tree so that only minimal hitting sets are found and the number of invocations of the reasoning component is kept small. The method as stated by Reiter is:

> (1) Generate the pruned HS-tree breadth first, generating nodes at any fixed level in the tree in left-to-right order.
>
> (2) Reusing node labels: If node n is labeled by a set $S \in F$, and if n' is a node such that $H(n') \cap S = \{\}$, then label n' by S. Such a node n' requires no access to F. The label of node n' is underlined to indicate that it is a reused label.
>
> (3) Tree pruning:
> (i) If node n is labeled by $\surd$ and node n' is such that $H(n) \subseteq H(n')$, then close the node n'. A label is not computed for n' nor are any successor nodes generated. A closed node is denoted by $\times$.
> (ii) If node n has been generated and node n' is such that $H(n') = H(n)$ then close node n'.
> (iii) If nodes n and n' have been respectively labeled by sets S and S' of F, and if S' is a proper subset of S, then for each $\alpha \in S - S'$ mark as redundant the edge from node n labeled by α. A redundant edge, together with the subtree beneath it, may be removed from the HS-tree with preserving the property that the resulting pruned HS-tree will yield all minimal hitting sets for F.
>
> A redundant edge in a pruned HS-tree is indicated by cutting it with ")(".

In the context of the diagnostic process, an access to F in the HS-tree algorithm is an invocation of the inference mechanism. When a label for a node n must be computed (that is, the node cannot be closed or relabeled) then the underlying reasoning component must return one of two values. If there exists a conflict set S such that

$H(n) \cap S = \{\}$, then the reasoning component must return S otherwise the value $\surd$ is returned. Thus the reasoning component is passed the set COMPONENTS – $H(n)$ as well as the system description and observations. If SD $\cup$ OBS $\cup$ $\{\neg AB(c) \mid c \in$ COMPONENTS – $H(n)\}$ is consistent, then $\surd$ is returned. Otherwise, a conflict set (not necessarily minimal) is returned. It should be noted again that the underlying reasoning component must be a decision procedure for determining consistency. In general, such decision procedures do not exist. However, in some domains, decision procedures do exist.

Consider the example of the full adder under the set of observations specified. An HS-tree for the example is shown in Figure 3. Note that the single fault diagnoses are determined by the nodes labeled with $\surd$ at level 1 in the tree. If diagnoses of cardinality k or less are desired, then the construction of the HS-tree can be halted as soon as level k is complete. Thus, the single fault assumption which is prevalent in diagnostic work fits in well with the HS-tree.

When multiple diagnoses are indicated by the observations, it is desirable to reduce the number of diagnoses by taking additional measurements on the system. Genesereth [*Ge*84] presents a method for suggesting measurements. Part of that method involves the use of the novelty check which was discussed in the section on Genesereth's work. The validity of the novelty check can be demonstrated within the context of Reiter's theory. Reiter states and proves a theorem which is a slightly stronger version of the novelty check

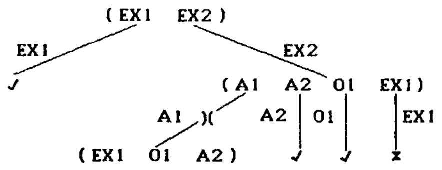
Theorem: Suppose every diagnosis for (SD, COMPONENTS, OBS) predicts one of Π, $\neg\Pi$. Then:

1) Every diagnosis for (SD, COMPONENTS, OBS) which predicts Π is a diagnosis for (SD, COMPONENTS, OBS $\cup$ $\{\Pi\}$).

2) No diagnosis for (SD, COMPONENTS, OBS) which predicts $\neg\Pi$ is a diagnosis for (SD, COMPONENTS, OBS $\cup$ $\{\Pi\}$).

3) Any diagnosis for (SD, COMPONENTS, OBS $\cup$ $\{\Pi\}$) which is not a diagnosis for (SD, COMPONENTS, OBS) is a strict superset of some diagnosis for (SD, COMPONENTS, OBS) which predicts $\neg\Pi$. In other words, any new diagnosis resulting from the new measurement Π will be a strict superset of some old diagnosis which predicted $\neg\Pi$.

The difficulty of reducing the number of diagnoses is illustrated by point three of the theorem. While it is true that measurements can confirm or reject diagnoses, the number of diagnoses can increase. Further, the number of components involved in a particular diagnosis can also increase. Except in the case where a measurement only rejects existing diagnoses, new diagnoses can arise. Reiter raises important research questions with respect to measurements. Is it possible to identify which measurement(s) will simply filter existing diagnoses and not result in any new diagnoses? The answer to this open question would be a significant result for diagnosis from first principles. A second problem is to make use of the existing HS-tree after a measurement is made. Because of the amount of computation necessary to build an HS-tree, it would be desirable to make use of known information, if possible, when rejecting existing diagnoses or computing new diagnoses.



Pruned HS-tree for the Fulladder

Figure 3

3. Boolean Term Rewriting Systems

Reiter's theory of diagnosis requires the presence of a sound and complete theorem prover in order to construct the HS-tree. Since the representation language for the circuit descriptions is first order logic and the circuits are Boolean, the refutational theorem proving system developed by Hsiang [*Hs*85] is well suited for this problem. Before proceeding to the implementation of Rieter's diagnostic model, the necessary terminology and methods of term rewriting systems and refutational theorem proving need to be discussed.

A *term rewriting system* is a finite set of directed equations of the form $l \rightarrow r$, which we call *rewrite rules* or reductions. Here l and r are terms and the equality is directed by some ordering scheme. A term t can be *reduced* by $l \rightarrow r$, if there is a subterm s of t and a substitution σ such that $l\sigma = s$. The reduction is made by replacing s with $r\sigma$. A term is *irreducible* or *in normal form* if no rule can reduce it.

Of particular interest are rewriting systems which possess the properties of (1) finite termination - after a finite number of applications of rewrite rules, an irreducible term is produced and (2) unique termination - if a term t can be reduced in two ways to terms r and s, then the normal form of r and s are the same. Term rewriting systems which satisfy these properties are called canonical rewriting systems or complete sets of reductions.

Unfortunately, there does not exist a complete set of reductions for Boolean algebra using the usual operations of AND, OR, and NOT. However, Hsiang succeeded in showing that by representing Boolean algebra formulas using the Boolean operations of AND ($\wedge$) and EXCLUSIVE-OR ($\oplus$) that indeed there does exist a complete set of reductions for Boolean algebra. As a consequence of this representation, every formula in the propositional calculus has a unique normal form (i.e., is either 0, 1, or equals $t_1 \oplus \ldots \oplus t_n$, where the t_i are products of distinct positive literals). These rewrite rules are:

$$\begin{aligned}
&x \oplus 0 \rightarrow x\\
&x \oplus x \rightarrow 0\\
&x \wedge 1 \rightarrow x\\
&x \wedge x \rightarrow x\\
&x \wedge 0 \rightarrow 0\\
&x \wedge (y \oplus z) \rightarrow x \wedge y \oplus x \wedge z
\end{aligned}$$

It should be noted that the operators $\wedge$ and $\oplus$ are both associative and commutative with identity 1 and 0 respectively. We will normally indicate the $\wedge$ operation by juxtaposition of the operands. The other Boolean operations can be converted into this format using the rules:

$$x \Rightarrow y \rightarrow xy \oplus x \oplus 1$$
$$x \vee y \rightarrow xy \oplus x \oplus y$$
$$\neg x \rightarrow x \oplus 1$$
$$x \Leftrightarrow y \rightarrow x \oplus y \oplus 1$$

Here 1 stands for true and 0 for false. These rules can be used to transform Boolean terms into their EXCLUSIVE-OR and AND equivalents, however, they are very inefficient. Hsiang gives a more direct method for effectively carrying out this transformation.

4. A Refutational Strategy

It is well known from predicate calculus that a formula A is valid if and only if its negation is unsatisfiable. Thus in order to prove that a given formula A is valid, one assumes the negation is satisfiable and seek a contradiction (i.e. a refutation). Whereas, most refutation theorem provers are modeled on the resolution principle [*Rb*65], Hsiang's theorem prover is based on the Knuth-Bendix completion procedure [*KB*70] which attempts to find a canonical set of rewrite rules in some existing rule set.

The central idea behind these methods is to convert Boolean formulas into Boolean rewrite rules and then apply certain superposition techniques to produce new rules from them. At all times, terms are reduced using the canonical rewriting system for Boolean algebra. This process continues until the contradictory rule $1 \rightarrow 0$ is generated indicating inconsistency or no more rules can be generated indicating consistency. A third possibility which arises in a general theorem proving environment is that the superposition process will run forever. This strategy is called the *N-strategy* and it and the superposition technique are completely described in [*Hs*85]. It should also be noted that this strategy requires the use of an extended unification algorithm called BN-unification which in this particular case is really an associative commutative identity unification.

An additional benefit of this particular approach is that it can also be used to verify the correctness of combinational circuits without the need of exhaustive simulation. For a discussion of this design verification system the reader should consult [*Ch*87].

5. Implementation and Examples

We have implemented a diagnostic system based on Reiter's theory of diagnosis. The underlying theorem prover is an implementation of a refutational theorem prover based on the work of Hsiang. Demodulation has been added to the rewriting techniques. The system is written in Common Lisp and been run on DEC MicroVax, Xerox 1109, and Symbolics workstation systems.

Reiter's theory is independent of the implementation of the underlying inference mechanism. The only requirement is that the inference mechanism be a decision procedure for the domain of the diagnostic problem. In general, a refutational theorem prover is a semi-decision procedure. However, in the domain of boolean circuits and with the addition of demodulation to the rewriting techniques, the inference mechanism is a decision procedure.

Our initial implementation did not make use of additional simplifiers (i.e. demodulators), but simply converted the first order logic descriptions of observations, components and system description into Boolean rewrite rules. This caused the theorem prover to be inadequately focused and as a consequence, it failed to determine consistency or inconsistency in a reasonable length of time. In order to focus the refutation process, we added the connection descriptions, the observed input and output values, and the logical operations as special demodulators which do not directly take part in the N-strategy of Hsiang. They are used instead to propagate values through the circuit and aid in the simplification of the Boolean rules which are generated as part of the N-strategy. It should be noted that under certain conditions a new rule can be used to form a new demodulator. For example, if the output of the gate EX1 is found to be 1 in the full adder, then ((OUT EX1) 1) is added to the list.

The procedure for building the hitting set tree uses all of the pruning techniques suggested by Reiter. These are the reuse of node labels, closing of nodes, and the removal of redundant edges. The last is not necessary if the inference mechanism returns only minimal conflict sets. We have no such guarantee of minimality, however. In addition, we have implemented two heuristics which can further decrease the number of conflict sets which must be computed.

Reusing the label of an existing node in the HS-tree as the label for a new node saves a call to the underlying theorem prover. As the HS-tree grows, there may be several labels which could be reused. The label with the smallest number of elements is chosen. This can be a valuable heuristic since every element of the set labeling a node generates a node at the next level. Further, after a conflict set is returned by the inference mechanism, a check is made to determine whether that conflict set could be used to label an existing node which has not yet been expanded. If the new label would have fewer elements, the node is relabeled.

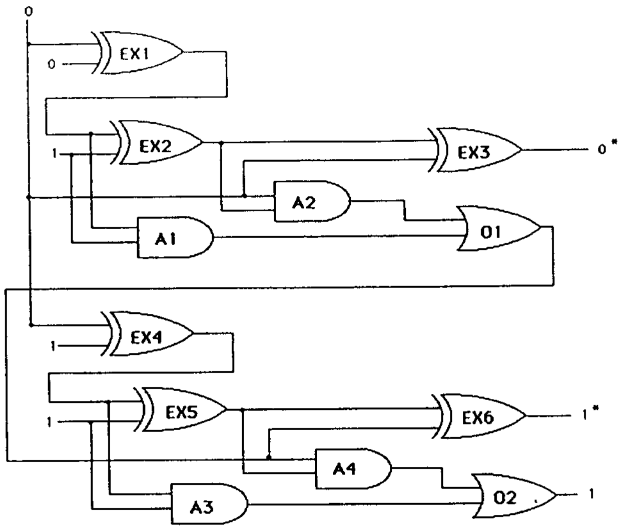
Our experience has shown that the removal of redundant edges usually does not decrease the number of calls to the theorem prover. Typically, those nodes which are removed by the pruning are closed or can be relabeled and thus do not require that a conflict set be computed. Whether this is an anomaly of the circuit examples we have studied or a feature of the general diagnostic domain is unknown. However, the overhead incurred by the removal of redundant edges is very small compared to the computation cost of even one unnecessary invocation of the theorem prover.

Consequently, our diagnostic program consists of essentially two parts. The first consists of those routines which implement the construction of the pruned HS-tree of Reiter, including the heuristics discussed above. The diagnoses of the given circuit will be read from the pruned HS-tree when it is completely constructed. A diagnosis is the collection of edge labels on the path from the root to a node in the HS-tree which is labeled by $\surd$.

The second major portion of this program is the theorem prover which in our case is based on rewriting rules for Boolean algebra. The theorem prover must determine whether the clauses are consistent or inconsistent and if inconsistent return a conflict set. As the N-strategy progresses and new rules are generated, an ancestor list for each rule is maintained. A conflict set represents those components for which $\mathrm{SD} \cup \mathrm{OBS} \cup \{\neg \mathrm{AB}(c) \mid c \in \mathrm{COMPONENTS}\}$ is inconsistent. When an inconsistency is found, the theorem prover returns the conflict set consisting of those components c for which the clause $\neg \mathrm{AB}(c)$ was involved in the refutation. Since we are only concerned with the discovery of conflict sets, the ancestor list for a rule consists only of those $\neg \mathrm{AB}(c)$ which are in the derivation of the rule. Thus, when the contradictory rule $1 \rightarrow 0$ is generated, the conflict set is the set of components in the ancestor list for the rule.

It should be noted that since our circuits are Boolean, we have added two axioms which assert that 0 and 1 are distinct constants. The first order logic representation is in clausal form, which means each input line is a disjunction of literals. Shown in Figure 2, is the description of the full adder circuit of Figure 1 and the resulting pruned HS-tree is illustrated in Figure 3. Furthermore, the first order logic description of the circuit must be preprocessed in order to put it into a clausal form based on the EXCLUSIVE-OR and AND operations necessary for the N-strategy.

We show in Figure 4 a more complex example which has been diagnosed using this technique.



TWO-BIT ADDER/SUBTRACTER

* - indicates incorrect value

DIAGNOSIS ((EX1) (EX4 EX3) (EX4 EX2) (A2 EX2) (A2 EX3)
(EX2 A1) (EX2 EX6) (EX2 EX5) (EX2 O1)
(EX3 A1) (EX3 EX6) (EX3 EX5) (EX3 O1))

Figure 4

REFERENCES

[*Ch*87] Chandrasekhar, M., Privitera, J. and Conradt, K., Application of Term Rewriting Techniques to Hardware Design Verification, *Proceeding of the 24th ACM/IEEE Design Automation Conference*(1987), 277-282.

[*Da*84] Davis, R., Diagnostic reasoning based on structure and behavior, *Artificial Intelligence* **24** (1984) 347-410.

[*dK*76] de Kleer, J., Local methods for localizing faults in electronic circuits, **MIT AI Memo 394** Cambridge, MA (1976).

[*DW*87] de Kleer, J., and Williams, B., Diagnosing multiple faults, *Artificial Intelligence*, **32** (1987) 97-130.

[*Ge*82] Genesereth, M., Diagnosis using hierarchical design models, *Proceedings of the National Conference on Artifical Intelligence*,Pittsburgh, PA (August, 1982) 278-283.

[*Ge*84] Genesereth, M., The use of design descriptions in automated diagnosis, *Artificial Intelligence* **24** (1984) 411-436.

[*Hs*85] Hsiang, J., Refutational Theorem Proving using Term-Rewriting Systems, *Artificial Intelligence*, **25** (1985) 255-300.

[*KB*70] Knuth, D. E. and Bendix, P. B., Simple Word Problems in Universal Algebras, in *Computational Algebra*,J.Leach,(ed.), Pergamon Press, 1970 263-297.

[*Ku*87] Kuipers, B., Qualitative simulation as causal explanation, *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-17** 3 (1987) 432-444.

[*Re*87] Reiter, R., A theory of diagnosis from first principles, *Artificial Intelligence*, **32** (1987) 57-95.

[*Rb*65] Robinson, J. A., A Machine Oriented Logic based on the Resolution Principle, *J. ACM*, **12**, 1(1965) 23-41.