Sādhanā, Vol. 19, Part 1, February 1994, pp. 23–50. © Printed in India.

Fault diagnosis of machines

H N MAHABALA†, A T ARUN KUMAR, R R KURUP and G RAVI PRAKASH

KBCS Laboratory, Department of Computer Science and Engineering, Indian Institute of Technology, Madras 600036, India

E-mail: mahabala@iit.ernet.in

Abstract. This paper presents four major approaches for diagnosing machine faults. Given the description of a system to be diagnosed and the observations on the system when it works, the need for diagnosis arises when the observations are different from those expected. The objective of diagnosis is to identify the malfunctioning components in a systematic and efficient way. The four approaches discussed are based on fault-tree, rule, model, and qualitative model. Early diagnosis systems used fault-tree and rule-based approaches. These are efficient in situations where an expert is able to provide the knowledge in the form of associations between symptoms and faults. Model-based and qualitative model-based approaches overcome many of the deficiencies of the earlier approaches. Model-based approaches can take care of situations (faults) not envisaged *a priori*. Also, one can cater to minor variations in design using the same set of components and their interconnections. This paper discusses in each case, how the knowledge is represented and what diagnosis technique is to be adopted, and their relative advantages and disadvantages. Implementation of each method is also discussed.

Keywords. Rule-based diagnosis; fault-trees; model-based diagnosis; qualitative model-based diagnosis; power system diagnosis; multi-level qualitative reasoning.

1. Introduction

Diagnosis is one of the major application areas of knowledge-based systems today. If observations on a system in operation differ from the behaviour expected of the system, then the need for diagnosis arises. The goal of the diagnosis is to identify the malfunctioning components of the system.

There are many different approaches to diagnostic reasoning. Diagnostic fault-trees, rule-based reasoning, model-based reasoning, and qualitative model-based reasoning are the strategies used successfully in some narrow domains. Most of the early diagnostic systems used fault-trees (Williams *et al* 1983) and rule-based approaches (Vesonder *et al* 1983; Fukul & Kawakami 1986; Talukdar *et al* 1986).

Fault-trees provide a simple and efficient way to express the tests and conclusions

thereof needed to guide the diagnosis under various conditions. The diagnostic procedure traverses the tree starting from the root, applying the test at each node to decide the branch to be taken next, until it reaches the lead node (repair node).

In the rule-based approach, knowledge for diagnosis is captured in the form of IF-THEN rules. Rule-based systems are built by accumulating the experience of expert diagnosticians in the form of empirical associations between the symptoms of an abnormal system and the underlying faults.

Rule-based and fault-tree based approaches have some disadvantages in spite of their simplicity and efficiency. Both the systems are device-specific and must be reformulated even for a minor change in the device configuration. These systems are expensive to build and maintain. A small change in the device may require major restructuring of the tree or rules.

The model-based approach (de Kleer & Williams 1987; Struss 1988), the successor to the rule-based approach, attempts to overcome many of the limitations of the early systems. In the model-based approach, the device to be diagnosed is modelled and represented in terms of the structure and function of the individual components comprising the device. The correct behaviour of the device is inferred from the knowledge of the individual components and their interconnections. The model-based approach has the following advantages.

- The device description (function and structure) is represented explicitly;
- a domain-specific component library can be established;
- a device can be diagnosed by having a domain-independent diagnostic procedure which uses the model of the device derived from the library;
- it is possible to cover a large class of devices built out of the same set of components;
- diagnosis of new devices about which one does not have sufficient experience is possible;
- one can cover new symptoms and related faults.

A model-based approach requires a precise mathematical model of behaviour/functionality. But often, for diagnostic purposes, one can start with a description using trends and tendencies without resorting to precise quantification. The qualitative reasoning approach is based on such qualitative models and can often be used where models are not available or are too complex to deal with. The precision of the behavioural description in the quantitative models could be sacrificed by qualitative methods by retaining the crucial distinctions. Instead of continuous real-variables, each variable is described qualitatively using a small number of qualitative labels (e.g. $+, -$ or 0). Quantitative differential equations are converted into qualitative differential equations called confluences (de Kleer & Brown 1984).

Using a single model for troubleshooting in all situations may not work. There may be need for multiple models of the same device with different simplifying and operating assumptions. The qualitative model-based approach has the following advantages, in addition to those of the model-based approach.

- Adopts an approach followed traditionally by practising engineers to describe behaviour and express malfunctions;
- derives qualitative behaviours which are adequate and efficient from precise mathematical models;
- since fuzzy models are very similar to qualitative models, one can use results from the fuzzy systems area.

2. Diagnosis using fault-tree

Fault-trees are a natural and efficient way to represent a hierarchical organization of tests and conclusions thereof needed for the diagnosis of industrial equipment. Diagnosis can be viewed as the task of hypothesizing the locality of a fault and then successively refining the hypotheses based on the results of a hierarchy of tests at each step. A scheme that captures such knowledge naturally and effectively is the fault-tree. However, one is not satisfied with just the localization of the faults in most cases. The fault needs to be repaired. Repair information can be stored in the fault-tree to supplement the diagnosis. In our implementation, called IITMDESS (Mahabala *et al* 1992), we go a step further and provide back-pointers to the tests at the repair nodes. These back-pointers help in ascertaining the effectiveness of the repair done and in determining the existence of multiple faults.

A fault-tree consists of a set of different types of nodes to represent the diagnostic knowledge. Node types correspond to different situations which arise while performing diagnosis. The current implementation uses six different types of nodes: Control, Repair, Text, Branch, Or, and Subtree. These different nodes are shown in figure 1.

2.1 *Types of nodes*

Or node: A symptom is usually associated with a set of possible causes (by cause, we mean a faulty component/subcomponent). In some domains, it is a practice to order these causes statically in a priority order. In other domains, one may order the causes dynamically depending upon the parameters like MTBF (mean-time between failures), MRRP (most recently replaced part) etc. An Or node captures and represents this information. It represents the likely causes as branches to subtrees. If information on static ordering is available, it is represented as weights on the branches to subtrees.

Text node: In the course of locating a fault, one may want to separate a subsystem (disassemble) before continuing further. For example, one may want to disconnect a

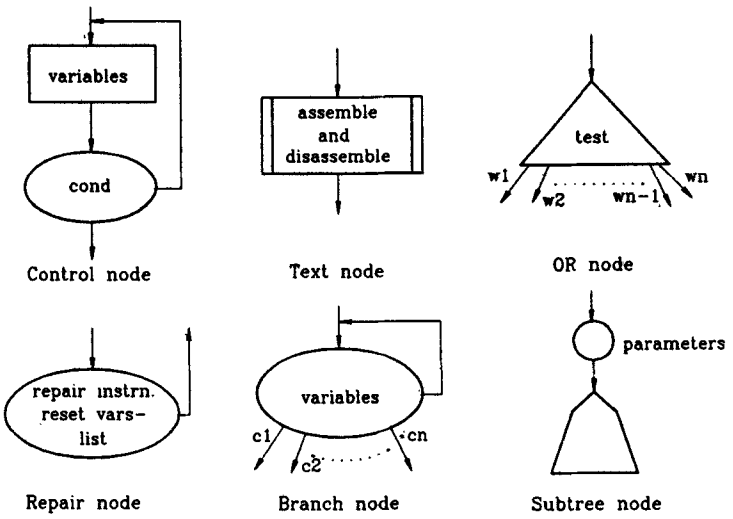

Figure 1. Different types of nodes of a fault tree.

power supply before diagnosing the equipment. Text node allows one to capture information that instructs the operator to disassemble sub-components on the way down and assemble them again on the return traverse.

Control node: This type of node decides whether there is a faulty condition of a certain category. It also specifies a list of variables which should be measured, and a conditional expression, which is evaluated to determine whether there is a fault of a specified type or not.

Repair node: When a diagnosis engine enters a repair node, one fault is identified. Repair nodes capture repair instructions, execution of which will eliminate the fault. Since repair may invalidate certain earlier measurements which need to be remeasured, one can also include a list of variables that should be reset after the repair action. One can associate a level of competence with a repair node which can be used to decide whether the current operator can handle the repair or not. The competence level is based both on training as well as access to tools needed. On completion of the repair, one has to go up in the fault-tree (rechecking certain measurements) to confirm the effectiveness of the repair, which is referred to as return traverse.

Branch node: During diagnosis, it is possible that multiple repairs are to be carried out to a single component. Branch nodes represent this knowledge. Branch nodes are similar to the control nodes, but they can have many child nodes. A branch node enables evaluation of multiple conditional expressions (one for each of its child nodes) and guides the diagnosis engine through one or more of its child nodes.

Subtree node: Complex equipment often have many components, subcomponents etc. Independent fault-trees can be developed for each subcomponent. A subtree node in the component level fault-tree enables diagnosis to enter an independent fault-tree corresponding to a subcomponent. This feature enables modular development of a fault-diagnosis system.

2.2 *The inference mechanism*

The inference mechanism consists of two steps. In the first step, we search for the cause of the abnormal behaviour of the machine in terms of repair needed. In the second step, after the repair has been done, a return traverse is used to check whether the problem has been completely eliminated. The basic reasoning cycle of the fault-tree based inference engine is described by the flow chart given in figure 2.

The processing starts from the root node traversing down dictated by the measurements and the type of nodes of the tree. When an Or node is entered, the system asks whether the corresponding symptom is observed. If so, the successor with the highest priority is selected for processing. Otherwise, the search backtracks to the parent node.

When a control node is entered, then the conditional expression associated with that node is evaluated using the values of the variables (which are measured if not already done before). If the condition is satisfied, then the successor node is explored.

When a branch node is reached, values are measured and the conditional expressions associated with the branch node are evaluated. If a variable has been measured earlier, it will not be measured again. The user is prompted whenever a measurement is

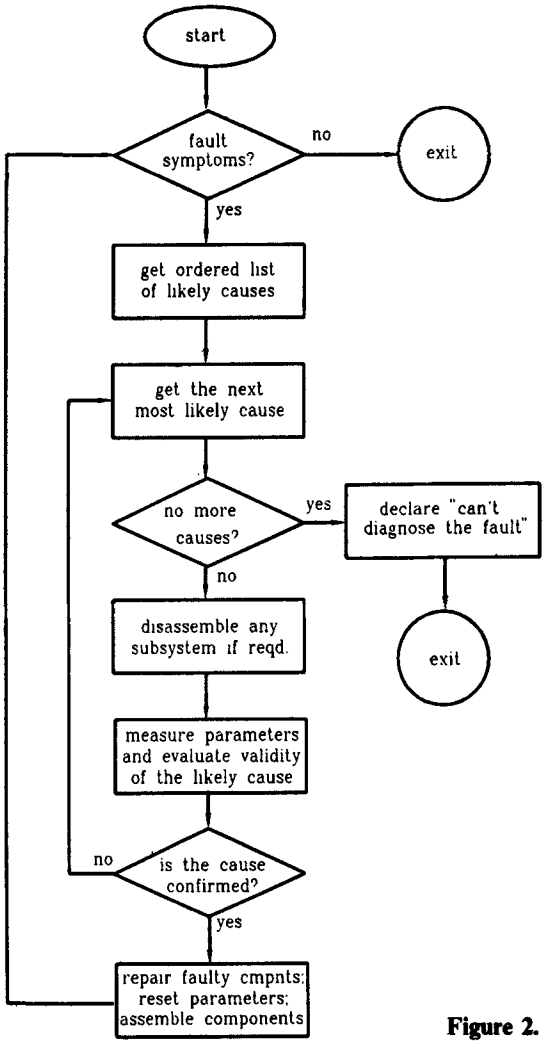

Figure 2. Reasoning cycle of the inference engine.

needed. If only one condition holds true, the corresponding branch is entered. If more than one condition holds true, its corresponding successors are processed in a left to right order or in the order of decreasing weights. If none of the conditions are satisfied, then the process backtracks to the parent node, deducing that the fault is elsewhere, and other paths are explored.

When a repair node is reached, the repair instructions are displayed by the system. After each repair, the parameters included in the list of resetable variables are reset and the search backtracks to the parent node. There is a provision to jump to a level higher than the parent, thereby making the checking of the elimination of fault more efficient.

While searching, if a subtree node is encountered, then during forward traverse the subsystem fault-tree is rolled in and the relevant parameters are passed to the subtree. The diagnosis continues from the root of the subtree. During return traverse, the subtree is rolled out and backtracking continues in the parent tree.

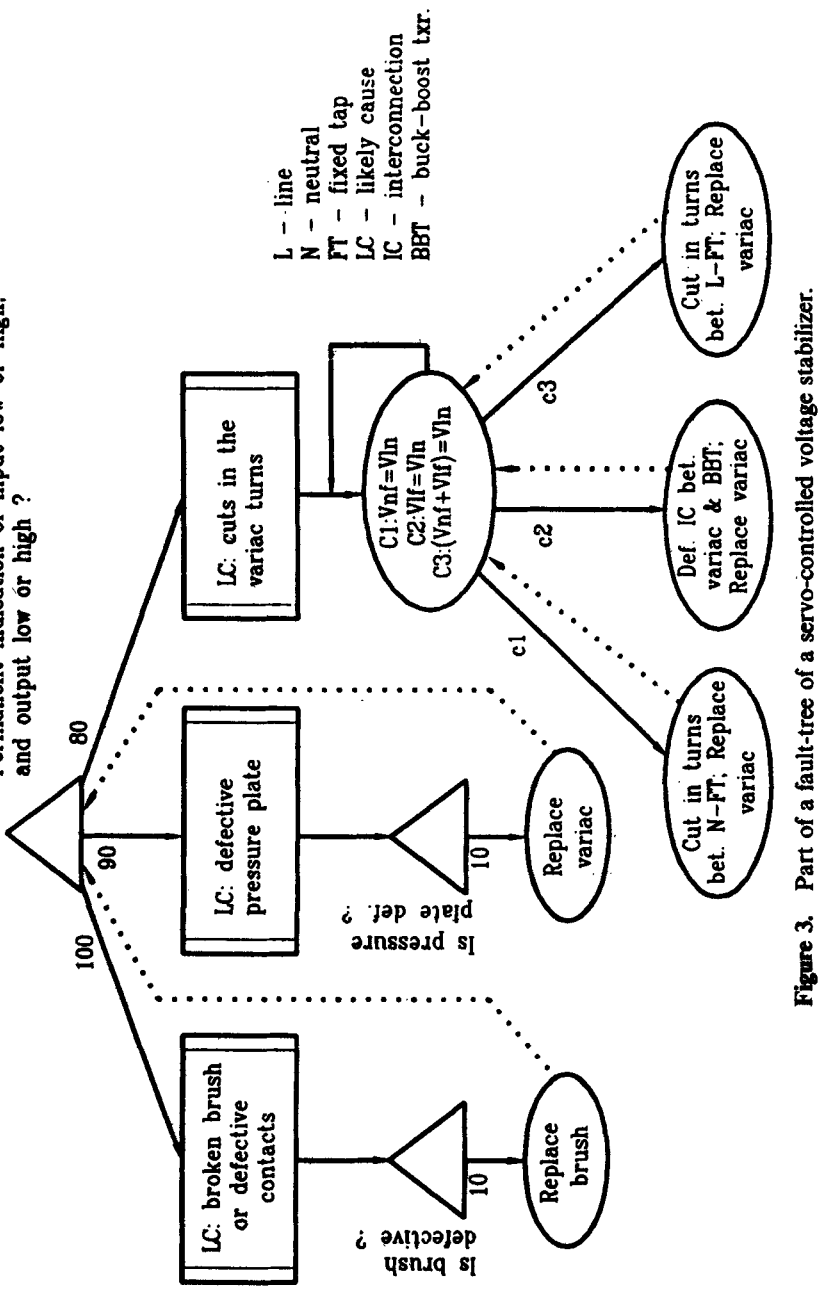

Figure 3. Part of a fault-tree of a servo-controlled voltage stabilizer.

2.3 *Example*

In this example, we illustrate the diagnosis of a particular brand of servo-controlled voltage stabilizer. A part of the fault-tree is shown in figure 3. Let us assume that there is a cut in the turns between the neutral and the fixed-tap of the variac. The diagnosis process starts from the root node and prompts the user whether there is any abnormality in the input-output panel indications. As the answer will be 'yes', the inference mechanism first checks whether the cause is the faulty brush. Having found that the cause is not the brush, the system verifies the cause is the faulty pressure-plate of the variac.

On obtaining a negative answer, the inference process verifies whether the cause is any cut in the variac taps. The system seeks values to the parameters, Vln, Vnf and Vlf. The three conditional expressions of the branch node are evaluated. Since we assumed that there is a cut in the turns between the neutral and the fixed tap of the variac, the system finds that (Vnf = Vln) and instructs the necessary repair. A part of the diagnosis session corresponding to this fault is shown in appendix A.

2.4 *Special features of IITMDESS*

The following are the special features of IITMDESS.

(1) The system has a knowledge-acquisition module which provides an interactive environment for the expert to graphically edit the fault-trees;
(2) the system has a facility to suspend the diagnosis session at any time and to resume at a later time. This feature helps the repairs, which take a long time, by relieving the computer for other uses;
(3) the system makes use of the same fault-trees in providing a practically useful 'Training and Testing Facility' to the service engineers.

3. Rule-based diagnosis

Rule-based approach is an efficient and simple way of organizing the knowledge of a machine to be diagnosed in the form of rules. A rule takes the following forms:

IF ⟨antecedent-1⟩
AND ⟨antecedent-2⟩
.....
AND ⟨antecedent-*n*⟩
THEN ⟨consequent-1⟩
AND ⟨consequent-2⟩
.....
AND ⟨consequent-*m*⟩

Rules incorporate knowledge which relate symptoms with the underlying malfunction. Traditional rule-based systems have been built by accumulating the experience of expert diagnosticians in the form of rules. The association of symptoms with the underlying faults are based on the experience with the device to be diagnosed. For example, a fault in the motor-starting system would be represented as:

IF the engine does not turn over
AND the battery voltage is ok
AND the starter motor relay operates
AND the starter motor does not turn
THEN the starter motor is defective.

A rule can also conclude an intermediate condition (hypothesis). The rules are organised such that one or more intermediate hypotheses are deduced and then combined to produce the final diagnosis of the system.

The diagnosis starts with an observation (or problem) of the working system. Based on the problem, rules are selected and fired, gathering new observations until either a faulty component is diagnosed or the fault cannot be diagnosed with the available knowledge (rules). Generally, the user is queried for new observations/measurements as the diagnostic session progresses. When it identifies a faulty component, repair action or similar advice could be suggested. In the case of multiple faults, the diagnostic process can be continued after the repair until all faults are identified and repaired. One should adopt backward chaining (goal-driven) to make the system prompt the user. If the results of all tests are available at the start one can use forward chaining (data-driven).

A portion of the rule-base for PC diagnosis (for floppy disk problems) and a sample diagnosis session are given in the appendix B.

3.1 *Implementation*

A rule-based shell, IITMRULE, developed by the nodal centre on expert systems at our Institute has been used to implement the diagnostic system. IITMRULE is a powerful tool for building rule-based systems. It has a rule base editor and a compiler. It supports backward chaining as well as forward chaining of rules. Also, it has a database toolkit to access a database. The detailed description can be found in Mahabala & Ravikanth (1990).

4. Model-based diagnosis

Suppose one has an adequate model (usually mathematical) of a system. If there is a difference between the behaviour manifested by the system and that predicted by the model, the system is faulty and we need to identify the fault(s). The diagnostic task is to identify the faulty components which explain the discrepancy between the observed and the expected behaviour of the system. The general principles of the model-based diagnosis are shown in figure 4. The model-based diagnosis starts with:

- SD, a description of the system to be diagnosed given in terms of the model for the normal behaviour of its components and their interconnections, and
- OBS, a set of observations about the real system.

From the model and an initial set of values, the diagnostic system predicts the expected behaviour of the system. If all the components are functioning correctly, then SD $\cup$ OBS $\cup$ COMP is consistent, where COMP is the set of assumptions about the correctness of the components. If SD $\cup$ OBS $\cup$ COMP is inconsistent, then the need for diagnosis arises. The diagnoses Δ is a set of components, $\Delta \subseteq$ COMP, such that

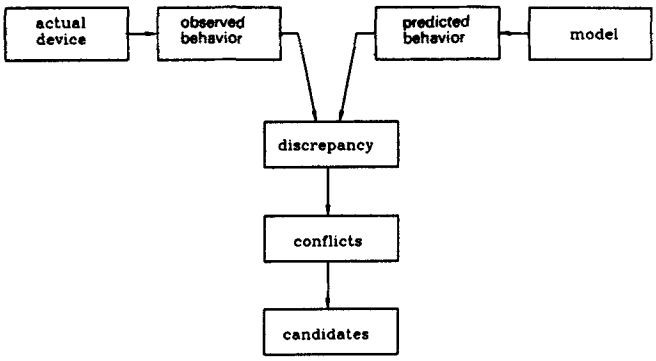

Figure 4. Principles of model-based diagnosis.

SD ∪ OBS ∪ COMP* is consistent, where COMP* is obtained by replacing components in Δ by their faulty models.

The diagnostic process is incremental. Diagnoses are determined by generating prediction about observable parameters of the device from subsets SD′ ⊆ SD and COMP′ ⊆ COMP and then checking these predictions for consistency with the observations OBS′ ⊆ OBS. It is possible to gather evidence against correctness assumptions of the components by analysing the root-cause of discrepancies between predictions and observations. From this discrepancy, a set of components called conflict is generated. Conflict is a set, CONFL ⊆ COMP, such that some or all components in CONFL are faulty. That is, all the components in the conflict set cannot work correctly. More observations would produce more conflicts, and the set of all the conflcts is referred to as CONFLICTS. From the conflicts, sets of components called candidates (diagnoses) are generated. A candidate is a set CAND ⊆ COMP such that: ∀ CONFL ∈ CONFLICTS [CAND ∩ CONFL ≠ Φ]. That is, a candidate must have at least one component from all the conflicts.

Usually, additional observations are necessary to isolate the set of components which are actually faulty. Each observation may produce new sets of conflicts and they are used to refine the candidates so far produced. The best next measurement is the one which will, on an average, lead to the discovery of the set of faulty components in a minimum number of measurements (de Kleer & Williams 1987).

An assumption-based truth maintenance system (ATMS) (de Kleer 1986) is used as a tool which records assumptions, inferences and their dependencies, observations, inconsistencies etc. The ATMS is a powerful tool for multiple context reasoning, and it is capable of working with multiple contexts simultaneously. This feature of the ATMS is used by the diagnosis system. Section 4.1 gives an overview of the basic ATMS, and §§ 4·2 and 4·3 discuss examples, algorithms, implementation and a practical application of the model-based diagnosis.

4.1 *Overview of the ATMS*

Model-based reasoning systems, such as diagnosis systems, make inferences based on certain assumptions. These inferences have to be recorded with their dependencies for further reasoning. Also, inconsistencies have to be detected as and when they occur, to prevent their propagation. An ATMS provides a good framework to achieve

this, and forms one of the components of the overall problem solving system (de Kleer 1986b; Dressler & Farquhar 1990). Every inference made by the problem solver is communicated to the ATMS. The task of the ATMS is to keep track of these inferences and the associated assumptions which support those inferences.

ATMS works with a set of nodes and a set of justifications, where a node is an internal representation for a problem-solver datum, and a *justification* is the reason based on which a node has been justified. A justification is of the form $a_1, a_2, \ldots, a_n \Rightarrow c$, where $a_1, a_2, \ldots, a_n$ are called *antecedents* and c is called *consequent*. A subset of antecedents are *assumptions*, whose truth or false value can get changed. For example, an assumption that a component works properly may have to be revised to false to explain a fault. Another subset of antecedents consists of *facts*, which are always true.

We refer to a set of assumptions as an *environment*. Each node in the ATMS is marked with a set of environments (any one justifies the node), referred to as *labels*. Each label is consistent and minimal. A label is consistent if all its environments are consistent. A label is minimal if no environment of the label is a superset of any other in that label. Inconsistent environments are called *conflicts*. A label of an assumption is a set of a set of the assumption itself. For example, label of the assumption B is $\{\{B\}\}$. A label of a fact consists of one empty environment. For example, the label of any fact is $\{\{\ \}\}$.

When a new justification is added, the ATMS computes a new label for its consequent node, and if there is a change in the new label, the new label is propagated through the network of justifications. The label of the node with respect to the new justification is the cross product of the labels of its antecedents. If a node has more than one justification, its label is the union of the labels contributed by those justifications. Consider the example in figure 5. The label of $n1$ due to $j1$ is $\{\{A, B\}\}$, and due to $j2$ is $\{\{D, E\}\}$. The combined label due to $j1$ and $j2$ is $\{\{A, B\}\{D, E\}\}$. The environment $\{C, D, E\}$ has been removed from the label of $n2$ because it is a superset of the conflict $\{C, D\}$. The final label of $n2$ is $\{\{A, B, C\}\}$. The label of the contradiction node $n3$ is $\{\ \}$.

4.2 *A model-based diagnosis example*

Consider the circuit in figure 6, which consists of two exclusive OR gates X1, X2, two AND gates A1, A2 and an OR gate O1. Assume the inputs are $A = 1$, $B = 1$ and $C = 0$. On the correct working of the circuit, the other values are $X = 0$, $Y = 0$, $Z = 1$, $F = 0$, and $G = 1$. Now the outputs are measured showing $F = 1$ and $G = 0$. From the measurements, it is possible to deduce that at least one of the following sets of

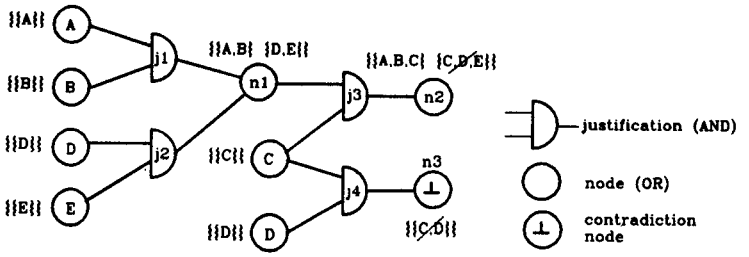

Figure 5. A dependency network showing nodes and justifications.

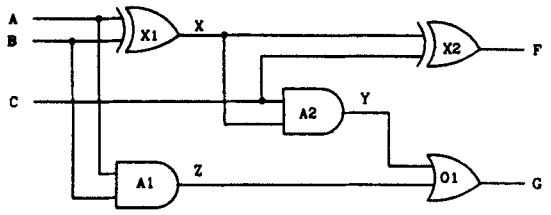

Figure 6. An example (full adder) for model-based diagnosis.

components is faulty:

[X1, A1], [X1, O1], [X2, O1], [X2, A1].

More measurements can isolate the faulty components. In the following discussions we assume that the interconnections are intact and hence they are not modelled explicitly. But they can be modelled as components with input equal to the output.

4.2a *Representing the system and its behaviour*: As mentioned earlier, the model of the system is a description of its physical structure plus models for each of its constituents. Each component is modelled as a set of constraints. A constraint is a relationship between variables. Consider the example in figure 6. A constraint $A \oplus B \Rightarrow X$ can be defined to derive X from A and B. If the values for A and B are known, the constraint fires and produces a value for X. For example, if $A = 1$ and $B = 0$, then the constraint produces $X = 1$. Similarly, F can be derived using the constraint $X \oplus C \Rightarrow F$. If we define both the constraints $A \oplus B \Rightarrow X$ and $X \oplus C \Rightarrow F$, the interconnection between X1 and X2 is automatically satisfied. If the values for A, B and C are known, then the first constraint (for X1) produces a value for X and the second constraint (for X2) produces a value for F. The constraints described above for X1 and X2 form the partial models for the components X1 and X2 (the complete models are described in the next subsection). Once the complete models for all the components are defined, the description of the device must be complete.

4.2b *Deriving the behaviour*: The behaviour of the system is generated by a method called *constraint propagation*. Constraint propagation operates on variables, values and constraints. Given a set of initial values, constraint propagation assigns each variable a value that satisfies the constraints. If sufficient values are known for a constraint, that constraint triggers and produces a new value, which in turn may trigger other constraints and so on.

The constraint propagation process derives new values by propagating known initial values through a set of constraints (which represents a set of components). Each derivation is recorded in the ATMS with its dependencies, which trace out a particular path through the constraints that the inputs have taken. For example, if $A = 1$ and $B = 1$, and A1 is working correctly (figure 6), then the constraint propagation produces $Z = 1$ and adds a justification: $A = 1$, $B = 1$, $A1 \Rightarrow Z = 1$, where A1 is the assumption about the AND gate A1, which shows that $Z = 1$ depends upon the correct functioning of A1. If we assume that $A = 1$ and $B = 1$ are facts, then $Z = 1$ receives a label $\{\{A1\}\}$. One advantage with this approach is that it does not differentiate between inputs and outputs. A path may begin at any point in the circuit where a measurement has been taken. Also, it is not necessary to make any assumption about

the direction of the signal flow. As a result, the component model should capture all the constraints such that any terminal (variable) value must be derivable, provided some other values are known. The complete model for the example in figure 6 is shown below.

A1:

$A \wedge B \Rightarrow Z, Z = 1 \Rightarrow A = 1 \wedge B = 1$

$Z = 0 \wedge A = 1 \Rightarrow B = 0, Z = 0 \wedge B = 1 \Rightarrow A = 0$

A2:

$X \wedge C \Rightarrow Y, Y = 1 \Rightarrow X = 1 \wedge C = 1$

$Y = 0 \wedge X = 1 \Rightarrow C = 0, Y = 0 \wedge C = 1 \Rightarrow X = 0$

X1:

$A \oplus B \Rightarrow X, B \oplus X \Rightarrow A, A \oplus X \Rightarrow B$

X2:

$X \oplus C \Rightarrow F, F \oplus X \Rightarrow C, F \oplus C \Rightarrow X$

O1:

$Y \vee Z \Rightarrow G, G = 0 \Rightarrow Y = 0 \wedge Z = 0$

$G = 1 \wedge Z = 0 \Rightarrow Y = 1, G = 1 \wedge Y = 0 \Rightarrow Z = 1$

Let us assume that the initial values are $A = 1$, $B = 1$ and $C = 0$. These values are stored as facts within the ATMS. Propagation produces $X = 0$, $Y = 0$, $F = 0$, $Z = 1$ and $G = 1$. Every derivation is stored in the ATMS, and is associated with a label $\{e_1, \ldots, e_n\}$, where e_i's are environments. The label shows from which set of components the value has been derived. In the following discussions we denote an assertion x with the supporting environments, $e_1, e_2, \ldots$ as $[x, e_1, e_2, \ldots]$. Now the database contains the following:

$[A = 1, \{\ \}]$	$[B = 1, \{\ \}]$
$[C = 0, \{\ \}]$	$[X = 0, \{X1\}]$
$[Y = 0, \{X1, A2\}]$	$[Z = 1, \{A1\}]$
$[F = 0, \{X1, X2\}]$	$[G = 1, \{A1, O1\}]$

4.2c *Conflict detection*: As shown in the previous section each derivation is labelled with a set of environments, where an environment is a set of correctness assumptions of the components. For example, the label $\{X1, X2\}$ of $F = 0$ shows that $F = 0$ is derivable only when the components X1 and X2 are working correctly. If the observed value differs from the predicted value, then the set of environments of the predicted value becomes inconsistent. An inconsistent environment is called a conflict. We denote a conflict as $\langle A_1, A_2, \ldots, A_n \rangle$ where A_i's are assumptions. For example, if F is measured to be 1 then $\{X1, X2\}$ becomes a conflict, $\langle X1, X2 \rangle$. It shows that X1 or X2, or both could be faulty. Since the label for a datum is minimal, we record only minimal conflicts, and its supersets are not explored.

4.2d *Candidate generation*: A candidate is a particular hypothesis which explains how the actual artifact differs from the model. A candidate is represented by a set of assumptions, indicated by $[A_1, A_2, \ldots]$. 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 a set of minimal candidates. The goal of candidate generation is to identify the complete set of minimal candidates.

Given no measurements, every component must be working correctly, and the single minimal candidate is [ ].

Candidates are generated from the conflicts. 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 new tentative 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 the example in figure 6. Initially there is no conflict, thus the minimal candidate is [ ]. The resulting database when $A = 1$, $B = 1$ and $C = 0$ is as shown in §4·2b. If F is measured to be 1, then [$F = 1$, { }] is added to the database ($F = 1$ is a fact). Propagation produces [$X = 1$, {X2}] and [$Y = 0$, {X2, A2}]. The inconsistency between [$F = 1$, { }] and [$F = 0$, {X1, X2}] produces a new minimal conflict ⟨X1, X2⟩. The minimal candidates are [X1] and [X2]. Next suppose we measure G to be zero. Propagation gives [$Y = 0$, {O1}] and [$Z = 0$, {O1}]. The symptom $G = 0$ but not 1 produces the conflict ⟨A1, O1⟩. The new minimal candidates are:

[X1, A1], [X1, O1], [X2, A1], [X2, O1].

The resulting database is shown below:

[$A = 1$, { }] [$B = 1$, { }]
[$C = 0$, { }] [$F = 0$, { }]
[$G = 0$, { }] [$X = 0$, {X1}]
[$X = 1$, {X2}] [$Y = 0$, {X1, A2}, {X2, A2}, {O1}]
[$Z = 0$, {O1}] [$Z = 1$, {A1}]

Further, if X is measured to be 0, it produces the new minimal conflict ⟨X2⟩. The propagation gives [$Y = 0$, {A2}]. The minimal candidates are [X2, A1] and [X2, O1]. Finally, measuring $Z = 0$ produces the conflict A1, and the final candidate is [X2, A1]. The final diagnosis is [X2, A1]. The malfunctioning components A1 and X2 explain the symptoms $F = 1$ and $G = 0$.

4.2e *Implementation*: The system is implemented using an ATMS and a constraint system. The resulting architecture is shown in figure 7. From the above discussions it is clear that the system has to record the inferences (behaviour) and the set of

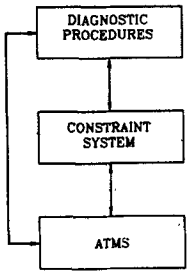

Figure 7. The architecture of the model-based diagnosis system.

assumptions under which they are true. An inference may depend upon other inferences. Thus the system is dealing with a network of inferences and their dependencies. The ATMS is a good tool for recording these inferences and dependencies in an efficient way. The current implementation of the ATMS (Mahabala & Kurup 1991a) provides a flexible interface with the application program. Now the bulk of the problem solving can be organized within the ATMS using consumers (de Kleer 1986b; Dressler & Farquhar 1990), where a consumer is a piece of code attached to a node which does some problem-solving work at the node.

The constraint system (Kurup & Mahabala 1992) provides functions to represent definitions of components, to create instances of specific components, and to set values to variables. We can create a library of component definitions using the functions provided by the constraint system, and they can be loaded whenever the structure of the system needs to be defined. Whenever the structure changes, a component can be created appropriate for the structure from the library of definitions. Thus, this approach can save an enormous amount of work which would have been required if there was no facility for establishing a library of components. Consider the example in figure 6. Let us assume that a library of components for digital circuits has been already established. Now the structure of the circuit can be defined by creating the specific instances for the components X1, X2, A1, A2, and O1 from the library. When the constraint system runs, it records the structure and behaviour (in the form of consumers) in the ATMS. That is, finally everything boils down to the ATMS. The details of implementation of the constraint system can be found in Kurup & Mahabala (1992). The conflict detection algorithm is implemented as part of the constraint system. The algorithm runs whenever a new observation is added or a new value is derived for a variable.

The top level diagnostic procedures include the commands to create the description of the system to be diagnosed and the candidate generation algorithm. It attaches the candidate generation algorithm to an ATMS node (called FALSE, which denotes contradiction) as a consumer. Whenever a conflict (nogood) is detected, the consumer runs and takes the conflicting assumptions as argument and produces the candidates. More details on the encoding techniques are discussed in Mahabala & Kurup (1991b).

4.3 *Diagnosis of a power system network. Practical application*

The purpose of a power system network is to distribute the electrical power from the generating sources to the customers in an efficient manner. The major components of the power system include the power sources (generators), transformers, connecting lines, buses, isolators and protective devices (circuit breakers, relays etc.). Normally the circuit breakers are closed and the lines carry power. When a fault occurs, the appropriate protective devices will operate and a portion of the network, which includes the fault, will be isolated. The effect of faults may disturb a large portion of the network or even the entire network. Problems can and do occur in protective equipment. These faults may cause tripping of many circuit breakers which results in a lot of information (messages) reaching a control centre. Automated fault localization based on the messages helps to restore the isolated network portions quickly and to ensure reliable distribution of power.

Since deriving rules from experience is very difficult, or even impossible, for a large network or a network which keeps on getting extended, it is necessary to use a model-based approach. A portion of a power system network (110/230 kV) is shown in

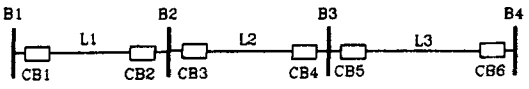

Figure 8. Portion of a 110/230 kV power system network.

figure 8. Presently we consider two types of protection: differential protection for buses, transformers and generators, and distance-relay protection for lines.

Assume that due to some disturbance (say a fault in B2) CB2 is tripped by the differential protection, and CB4 is tripped at distance level 2 (zone 2). The model-based approach produces the following diagnoses, one of which should be true.

(1) [B2, CB3] – bus B2 and the breaker CB3 could be faulty,
(2) [CB2, CB4] – breakers CB2 and CB4 could be faulty.
(3) [CB2, L2, CB3] – breaker CB2, line L2 (25% from CB3), and CB3 could be faulty.

Using additional information from the relays, the diagnosis system would isolate one fault – [B2, CB3]. We have implemented a power system network fault diagnosis system and tested on an actual 110/230 kV network (Kurup & Mahabala 1993).

5. Qualitative model-based diagnosis

The model-based approach for diagnosis given in §4 can provide more information. However, the computing cost is high and extensive knowledge about the device states and history is also needed. In certain cases, the theory or the rigorous understanding of the mechanisms involved may be lacking. The qualitative model-based approach will be able to provide diagnosis with a less rigorous understanding of the system's state and behaviour. In a qualitative model, one represents the values and nature of time behaviour of variables in qualitative terms, such as low, high, slow, fast etc. The input–output behaviour is also expressed in a qualitative manner, such as, if input goes up, the output goes down etc.

A single qualitative model for troubleshooting in all situations may not work. For example, diagnosing a faulty hydraulic circuit with a molecular level qualitative model of flow of liquids is difficult. A molecular level qualitative model of flow of liquids might be the right level to analyse the flow of molecules through a pump, regulator etc., but does not provide a useful qualitative model for diagnosing the pump, regulator etc. Hence, there is need for multiple models of the same device at various levels with different simplifying assumptions and operating assumptions. For analysing a device represented by multiple models at different levels, we should have the means to select the right qualitative model in a given situation. Information derived at a more abstract level, if any, should be used efficiently for analysis at the detailed levels.

The characteristics/requirements of the diagnostic system are:

- the system must be able to accurately diagnose both the single and multiple fault cases;
- due to interactions in a device, a fault in one component can propagate to other components; the system should be able to use this information;
- the system must reason about time because much information can be deduced from the sequence and time of events;
- a number of components combine to form a feedback loop. If there is a fault in any of the components, the fault can be magnified or compensated due to interactions

in the loop. The system must precisely determine the component responsible for the failure.

The approach to qualitative model-based diagnosis followed is the abduction-based approach as opposed to the consistency-based approach usually followed in model-based diagnosis.

The system to be diagnosed, D, is given by the pair ⟨COMP, MODEL⟩. MODEL is a description of the structure and behaviour of the system D. COMP is the set of components of D. The set of parameters producing information about the specific case under investigation is given by the context, CTXT.

Given the observed behaviour of the system at various time points, OBS = {OBSt1, OBSt2, OBSt3,...}, the diagnostic problem involves determining the modes of the components explaining the observations in CTXT, i.e., an assignment W for COMP is to be found such that $\forall$OBS$tx \in$OBS, (CTXT $\cup$ MODEL $\cup$ W) $\vdash$ OBStx. The model, MODEL, is represented at multiple levels of abstraction and approximation.

A framework for diagnosis with multiple levels of abstraction and approximation is presented in the following sections. Section 5.1 describes the architecture of the diagnostic system. Section 5.2 discusses an example, § 5.3 presents the multi-level qualitative model of the system described in § 5.2 and the multi-level qualitative reasoning. Section 5.4 gives a test case in which the solution to the diagnosis problem is given, which in turn gives the solution to the control problems of efficiency and selection in multi-level qualitative reasoning. Section 5.5 discusses the advantages of multi-level qualitative reasoning and § 5.6 gives the implementation status.

5.1 *Architecture of the qualitative model-based diagnostic system*

The architecture of the qualitative model-based diagnostic system is shown in figure 9. The justifications of the observations are found by multi-level qualitative reasoner

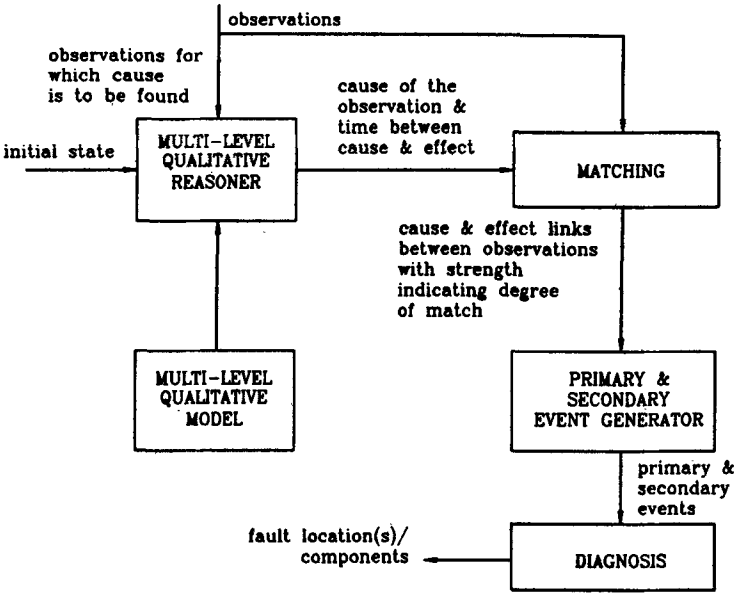

Figure 9. Architecture of the qualitative model-based diagnosis system.

using the behaviour generated from the multi-level qualitative model. The timing between the cause and the effect will also be generated. The justification for the observation derived from the qualitative model is verified among the set of observations for its presence by the matching module. The timing between the cause and effect derived from the qualitative model and the timing between the cause and effect actually observed are matched and a link strength based on the degree of match is formed between the cause and the effect by the matching module. The primary and secondary event generator generates the primary and secondary events based on the link strength. The diagnosis module generates the component(s)/location(s) responsible for the faulty behaviour from the primary and secondary events.

5.2 *Example*

The hydraulic system shown in figure 10 will be used to explain the concepts. The structural configuration of the system is as follows.

- pump with input *a* and output *b*;
- regulator with input *b*1 and output *c*;
- pipe between *b* and *b*1;
- filter with input *c*1 and output *d*;
- pipe between *c* and *c*1;
- electro hydraulic (EH) valve with inputs *d*1 and *d*2 and outputs *e* and *f*;
- pipe between *d* and *d*1;
- cylinder with input *e*1 and output *e*2;
- pipe between *e*1 and *e*;
- pipe between *e*2 and *d*2;
- pipe between *f* and sump;
- pipe between sump and *a*.

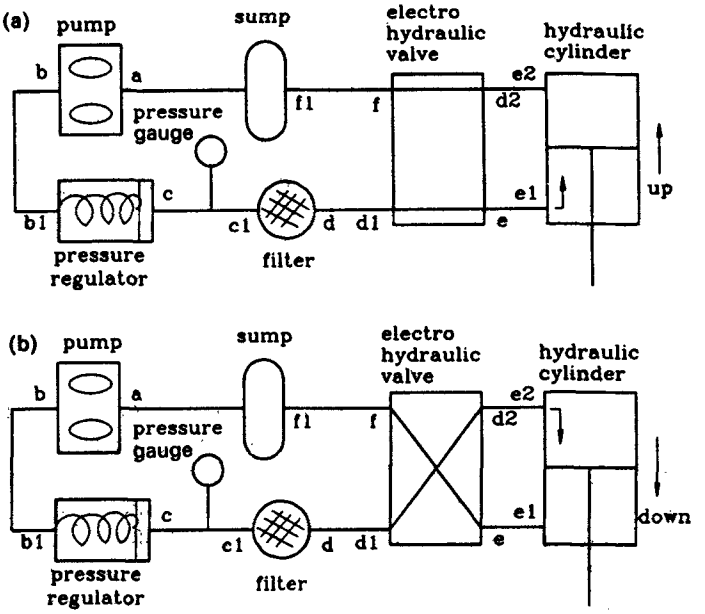

Figure 10. A hydraulic system during up (**a**) and down (**b**) of the EH valve.

Figures 10a and b show the system in two different positions (up and down traversal of piston rod) of the EH valve. The structural configuration for the system shown in figure 10b is the same as that for the system shown in figure 10a except that the EH valve has inputs $d1$ and e and outputs $d2$ and f.

The functioning of the circuit is as follows: Assume the EH valve is in the up-position as shown in figure 10a. The pump outputs the liquid at some pressure. The pressure regulator delivers liquid at a constant pressure irrespective of changes in pressure at its input (namely pump output). Any dirt particles present in the liquid are removed by the filter. The liquid then passes through the valve and reaches the cylinder. Due to the pressure differential across the two sides of the piston head in the cylinder, the piston head is lifted up and hence the load attached to the piston rod is lifted up. If the EH valve is in the down-position as shown in figure 10b, the piston head and the load attached to the piston rod are pushed down. Direction of piston movement (up or down) is controlled by the position of the EH valve. The fluid out of the cylinder empties into the sump.

Assume that there is a leak in the connecting pipe $d - d1$. There is a reduction in pressure that reaches the EH valve. This in turn causes a reduction in pressure that reaches the cylinder. This causes the load attached to the piston rod of the cylinder to go up very slowly or come down slowly depending on the position of the EH valve. The piston rod also does not move by the desired distance.

The reason for the reduction in the distance moved by the piston rod of the cylinder is due to a reduction in the pressure that reaches the cylinder. The reason for the reduction in pressure that reaches the cylinder is due to reduction in pressure that reaches the EH valve. There is no other justification for the pressure reduction at the input of the EH valve except for a leak in the connecting pipe $d - d1$ or a fault in the filter or a fault in the filter and a leak in the connecting pipe $d - d1$. Incidentally, this is the way an engineer would reason about the working of the correct and faulty systems in qualitative rather than quantitative terms.

The change in value of pressure at e could have been due to a fault either in the filter or in the pipe or both in the filter and the pipe. A component behaves in a faulty manner when its mode is not as desired. The time taken for a pressure signal to traverse from the input to the output of the component in various modes for both the filter and the pipe are inferred from the model. The time taken for a pressure signal to travel from the filter to the EH valve with the filter in the normal mode and the pipe in the abnormal mode matches with the timing observed. The connecting pipe $d - d1$ is identified as faulty.

5.3 *Multi-level qualitative reasoning*

In our system, the multiple models are based on the following abstractions:

- several components are combined into one. For example, even though a pump contains a lot of subcomponents, it can be considered as a component with a particular gain function in a certain level of analysis (or reasoning);
- the components are represented at different levels of detail. For example, a hydraulic circuit can be represented at three levels of detail. (i) Functional level where the valves are considered as switches which can be turned off or on, (ii) timing level where the components are modelled as a fluid resistance and fluid capacitance pair, (iii) flow level where the components and the interconnections are modelled in greater detail.

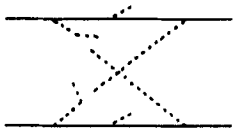

Figure 11. Figurative description of a qualitative functional model of the EH valve.

The hydraulic system shown in figure 10 will now be modelled at the three levels.

Functional level: At the functional level, the sub-components are combined and considered as a component with a particular gain function. The purpose of the model at this level is to check if the functioning of the functional blocks in the device is as expected. This checking is done considering the functional blocks in all possible modes. The general qualitative model of a component, *C*, at this level is:

status [relations], where status gives the mode of the component and the relations give the gain function.

The behaviour is analysed in terms of signals with order of magnitude values with the following quantity space [0, LOW, MEDIUM, HIGH, INF].

The EH valve is modelled as a collection of simple valves (normally closed and normally open) that can be turned on or off. The qualitative functional model of the EH valve is shown in figure 11. The modes of the EH valve are as shown in figures 10a and b. This model helps to answer questions regarding the liquid reaching the cylinder.

The pump can be modelled as a component with a gain function. For a given input-pressure, output-pressure = (input-pressure + pump-gain). For example, if input-pressure is LOW and pump-gain in MEDIUM, then output-pressure = (LOW + MEDIUM) = HIGH (derived). There are three modes of the pump where the gain is correspondingly zero, between zero and maximum, and maximum. The arithmetic followed is a simple order-of-magnitude arithmetic (Raiman 1986).

The cylinder can be modelled as a component with a gain function, where the distance moved by the piston rod is (input-pressure × cylinder-gain). The cylinder-gain depends on the distance available for the piston to move. The five modes of the cylinder are open up ↑, open down ↓, working up ↑, working down ↓ and stuck. The distance available for the piston to move is maximum in the first two modes, between zero and maximum in the next two modes and zero in the stuck mode.

The regulator can be modelled as a component with a gain function, where the output-pressure = (flow-rate × regulator-gain). The regulator-gain depends on the area available for the flow. The three modes of the regulator are open, closed and working. The area available for the flow in the above modes are maximum, zero, and between zero and maximum, respectively.

The filter can be modelled as a component, where the output-pressure = (input-pressure × filter-gain), where the gain depends on the area available for the flow. The three modes of the filter are open, closed and working. The area available for the flow in the above modes are maximum, zero, and between zero and maximum, respectively.

Timing level: At the timing level, the changes are that the components are more detailed with each component replaced by an *RC* pair (fluid resistance and fluid capacitance) in various modes of the components. The behaviour analysis involves two parameters *R* and *C*.

The purpose of the model at this level is to compute the time delay and check if

the observed delay between two components is as expected. Delay depends on the time constant of the path established by functional-level analysis. The time constants are calculated using the fluid resistance and the fluid capacitance in the path.

Flow level: At the flow level, the components and the interconnections are modelled in greater detail and the behaviour is analysed in terms of pressure and flow-rate. The purpose of the model is to check whether the components in the functional blocks behave as expected. The general qualitative model of the component C at the flow level is

modality [conditions] relations,

where the modality of a component is open, closed, working etc., and the [conditions] express the conditions that are true in the particular mode and the relations are the equations that hold in that mode.

There will be some bridging relations between the components at the functional level and the components at the flow level. The model of the bridging relation is expressed as:

component at the functional level;
components at the flow level;
relations.

For example, the filter and $d-d1$ pipe subsystem forms the functional level component, and its corresponding components at the flow level are the filter and the $d-d1$ pipe. Some of the relations existing between them are:

- the input of the component consisting of the filter and $d-d1$ pipe subsystem at the functional level is the input of the filter at the flow level;
- the output of the component consisting of the filter and $d-d1$ pipe subsystem at the functional level is the output of the $d-d1$ pipe at the flow level.

Given a domain described at multiple levels of abstraction, the approach taken is to start behaviour generation at the most abstract level and move down to a deeper level of reason only if the diagnostic problem demands it. Certain faults can also force the device into new modes of operation.

The algorithms for the reasoning at any level are given in Arun Kumar & Mahabala (1992). Several approaches to qualitative reasoning with multiple models have been made (Collins & Forbus 1987; Falkenhainer & Forbus 1988; Addanki *et al* 1989; Hibler & Biswas 1989; Liu & Farley 1990). Several approaches to qualitative reasoning with a single model have been that of de Kleer & Brown (1984), Forbus (1984) and Kuipers (1986).

5.4 *Test case*

The following test case illustrates the solution to the diagnosis problem using reasoning with multiple models. Consider the changes observed in the system behaviour over a period of time, which is shown in figure 12. In figure 12, $(t2-t1)$ is the observed time delay in the signal flow from $d1$ to $e1$ and $(t3-t2)$ is the observed time delay between the change at $e1$ and the change at the cylinder output.

The possible justifications for the observation at $t3$ (figure 12c) as generated by the QUALITATIVE REASONER (figure 9) at the functional level can be one of the following:

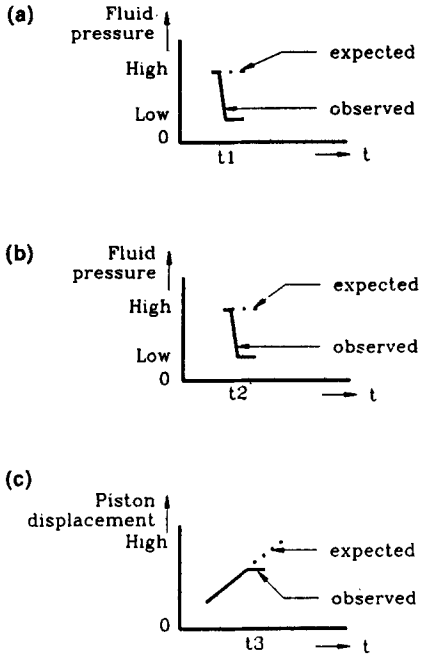

Figure 12. Observations of deviations in pressure at $d1$ (**a**) and $e1$ (**b**) and piston displacement (**c**).

(1) fall in pressure at $e1$ in mode open ↑ of the cylinder at an earlier instant of time;
(2) fall in pressure at $e2$ in mode open ↓ of the cylinder at an earlier instant of time;
(3) fall in pressure at $e2$ in mode working ↓ of the cylinder but slower than (2) at some earlier instant of time;
(4) fall in pressure at $e1$ in mode working ↑ of the cylinder but slower than (1) at some earlier instant of time.
(5) fall in flow-rate through the cylinder at some earlier instant of time.

The MATCHING module finds a change in pressure at $e1$ at $t2$ (figure 12b) among the set of observations. The cause for the change observed at $t3$ (figure 12c) could be one of (1) or (4). To select between (1) and (4) for the cause of the change observed at $t3$ (figure 12c), the timings in both the modes are generated at the timing level. The time taken for the pressure signal between cylinder input and output in mode (1) calculated using the model at the timing level and the observed timing between $e1$ and $e2$ match. Therefore, change in pressure at $e1$ at $t2$ (figure 12b) is established as the cause for the observation at the output of the cylinder at $t3$ (figure 12c) by the PRIMARY AND SECONDARY EVENTS GENERATOR (figure 9).

The possible justifications for observation at time $t2$ (figure 12b) generated by the QUALITATIVE REASONER at the functional level can be one of the following.

(1) Fall in pressure at $d1$ in the UP mode shown in figure 10a at some earlier instant of time;
(2) Fall in flow-rate across the EH valve at some earlier instant of time in the closed mode.

The MATCHING module finds the observation at $d1$ at time $t1$ (figure 12a). The timing between $d1$ and $e1$ calculated by the model at the timing level and the observed

timing between $d1$ and $e1$ also match. Therefore change in pressure at $d1$ at $t1$ (figure 12a) is established as the cause for the observation at the output of EH valve at $e1$ at $t2$ (figure 12b) by the PRIMARY AND SECONDARY EVENTS GENERATOR (figure 9).

The possible justification for observation at time $t1$ (figure 12a) generated by the QUALITATIVE REASONER at the functional level can be one of the following.

1. Fall in flow-rate through the filter at some earlier instant of time;
2. fall in pressure at $c1$ in mode open of the filter at some earlier instant of time;
3. fall in pressure at $c1$ in mode working ↑ of the cylinder but slower than 2 above at some earlier instant of time.

The MATCHING module does not find any of the above in the set of observations. Hence, there is no justification for the observation at time $t1$ (figure 12a). The fault is isolated to the subsystem consisting of the filter and connecting pipe $d-d1$.

Now, the fault has to be further localised. With the nominal values at $c1$ and $d1$, the modes of the components and the possible values at the intermediate points are generated by the QUALITATIVE REASONER at the flow-level. With the changed value at $d1$ and the nominal value at $c1$, the possible modes of the components and the possible values at the intermediate points are generated by the QUALITATIVE REASONER at the flow-level.

The modes which explain the change in the value at $d1$ are:

(1) the pipe is leaky;
(2) the filter is clogged; or
(3) the pipe is leaky and the filter is clogged.

Given a small change in pressure at the filter input, an observation is made on the time taken for the pressure change to reach the valve input. This is checked with the timing calculated from the model at the timing level in each of the three modes given above. The timings calculated from the model at the timing level in the case of the pipe, being leaky, match the observed timings. Thus the fault has been localised to the pipe being leaky. The diagram of the multi-level simulation process is shown in figure 13.

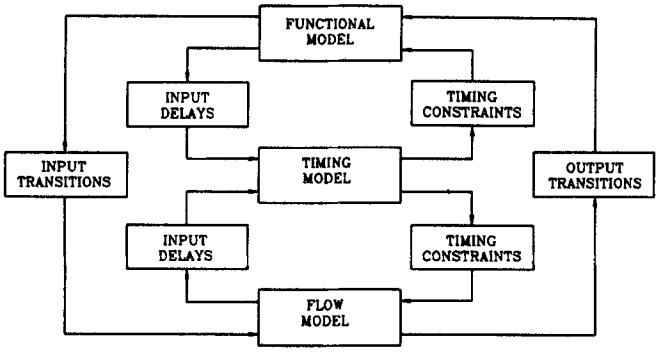

Figure 13. Interactions between models at different levels.

5.5 *Discussion*

It is difficult to simulate the circuit and test for all the three failure modes simultaneously. At the functional level, the failures associated with the functioning of each of the functional blocks are identified. After isolating the fault to a functional block, the fault location is localised to the subcircuits at the flow level in that block. Any ambiguity as to which of the components in the functional block is responsible while localising the fault at the flow level is resolved at the timing level. Thus, one of the advantages of using qualitative models at multiple levels for generation of the system behaviour is that the information derived at the higher levels is used for behaviour generation at more detailed levels, thereby making the analysis computationally less expensive.

In our framework, new knowledge is added by designing an appropriate new model which will work with the existing ones. For example, a timing model can be added to the functional model to increase the completeness of the description.

5.6 *Implementation*

Prototype Lisp programs have been written for simulation at functional, timing and flow level.

6. Conclusion

Diagnosis is a value-added area for the application of expert system technology. The traditional symptom-and-cause approach can be captured by the diagnosis tree or a set of rules. Required knowledge is based on experience and is referred to as causal model. One can expect wide use of the causal approach in years to come. The inflexibility of the causal model in terms of not being able to take care of even minor variations in the configuration can be overcome by the model-based approaches. The qualitative approach for describing behaviour, which often is the basis of design, can also be used to build a diagnostic system. Mechanical systems are generally handled by the use of qualitative models. The need for greater flexibility is provided by the model-based approach. One can handle a wide range of systems built using a set of components whose models are stored in a library. Power systems are suited for the model-based approach. Just as an engineer would first try to locate faults based on symptoms (causal model) and move to qualitative reasoning for more complex faults and finally resort to analytical approaches (model-based), so also one can build a multi-model diagnosis system which localises faults using causal models and see if faults can be further localised by qualitative models, and lastly resort to analytical models if warranted.

The authors would like to thank the Department of Electronics, Government of India and the United Nations Development Programme (UNDP) for assistance in carrying out this work.

Appendix A. A sample diagnosis session for a voltage stabilizer (see § 2·3):

Please input your name: A Raman
Please input your code: 12n45
Your competence is: 90
Please enter the Territory code: 300
Please enter the Customer Name: KBCS LAB
Please enter the Customer address: IIT MADRAS 600036
Please enter the Machine serial Number: 320
Please enter the Machine Rating (KVA/Phase): 430
Please enter the Model Name: SCVS, KRYKARD

(IITMDESS) starts

Is there any problem in the system?
(yes/no/suspend): yes

Is there output from stabilizer and any panel indication?
(yes/no/suspend): yes

Does the panel show permanent indication of output-low or output-high but not input-high or input-low?
(yes/no/suspend): no

Does the panel show permanent indication of input-low or input-high, and output-low or output-high?
(yes/no/suspend): yes

Likely cause: Variac brush broken or contacts defective.
{the system will find that the likely causes are not "broken brush" or "faulty pressure plate" of variac. Continuing with the diagnosis...}

Likely cause: Cut in the variac turns.
Check: With DMM set to 1000 V AC range, check voltage
between Line and Neutral, Line and Fixed Tap, Neutral and Fixed Tap.
Type 'done' when operation is complete,
'help' for help or 'suspend' to suspend diagnosis: done
Input integer value for Vnf: 230
Input integer value for Vlf: 120
Input integer value for Vin: 230

REPAIR to be done is the following

There is a cut in turns between Neutral and Fixed Tap
Replace Variac.
Type 'done' when repair is over
'help' for the help message or 'suspend' to suspend diagnosis: done

OLD PART REPLACED:

Name of the part: Variac
Type: 2
Serial No: 120

NEW PART REPLACED:

Name of the part: Variac
Type: 2
Serial No. 127

Does the panel show permanent indication of input-low or input-high, and output-low or output-high?
(yes/no/suspend): No

Is there any problem in the system?
(yes/no/suspend): No

Diagnosis Over
Please input your remarks for the current session:
VARIAC IS REPLACED.

Appendix B. A portion of the rule-base for the PC diagnosis (§ 3):

Rule for checking floppy is:
If there is a floppy disk problem
And the disk reads correctly
And the disk does not write
Then check other aspects
The rules for checking other aspects are given below.
R1:
If disk does not appear ok
Then the problem is due to physical damage of the disk.
R2:
If disk does appear ok
And the write protect hole on the disk is covered
Then problem is due to the write protect switch.
R3:
If disk does appear ok
And the write protect hole on the disk is not covered
And problem does not persist after removing the resident programs
Then problem is due to the resident programs.
R4:
If disk does appear ok
And the write protect hole on the disk is not covered
And problem persists after removing the resident programs
And the system has a second disk drive or access to another compatible computer

And the disk works the same way on the other drive
Then problem is due to a bad disk.

R5:

If disk does appear ok
And the write protect hole on the disk is not covered
And problem persists after removing the resident programs
And the system has a second disk drive or access to another compatible computer
And the disk does not work the same way on the other drive
And the disk loads properly and the motor turns after cleaning the read head
Then problem is due to dirt on the read head.

R6:

If disk does appear ok
And the write protect hole on the disk is not covered
And problem persists after removing the resident programs
And the system has a second disk drive or access to another compatible computer
And the disk does not work the same way on the other drive
And the disk does not load properly and the motor does not turn after cleaning the read head
Then problem is due to bad floppy drive.

R7:

If disk does appear ok
And the write protect hole on the disk is not covered
And problem persists after removing the resident programs
And the system does not have a second disk drive or access to another compatible computer
And the disk loads properly and the motor turns after cleaning the read head
Then problem is due to dirt.

R8:

If disk does appear ok
And the write protect hole on the disk is not covered
And problem persists after removing the resident programs
And the system does not have a second disk drive or access to another compatible computer
And the disk does not load properly and the motor turns after cleaning the read head
And it happens with only one program
Then problem is due to difficulty of some programs working with some disks.

R9:

If disk does appear ok
And the write protect hole on the disk is not covered
And problem persists after removing the resident programs
And the system does not have a second disk drive or access to another compatible computer
And the disk does not load properly and the motor turns after cleaning the read head
And it does not happen with only one program
And it happens with one particular floppy
Then problem is due to a bad disk or a particular brand of disks.

*R*10:

If disk does appear ok
And the write protect hole on the disk is not covered
And problem persists after removing the resident programs
And the system does not have a second disk drive or access to another compatible computer
And the disk does not load properly and the motor turns after cleaning the read head
And it does not happen with only one program
And it does not happen with one particular floppy
Then problem is due to bad floppy drive.

Typical diagnosis session: Let us assume that the floppy disk problem is due to a bad disk. Now a possible diagnosis session would be the following:

Q: Is there any floppy disk problem? (y/n)
A: y

Q: Does the disk read correctly? (y/n)
A: y

Q: Does the disk write correctly? (y/n)
A: n
(Now the rules for other aspects will be checked).

Q: Does the disk appear ok? (y/n)
A: y
(Now R1 fails and R2 is selected).

Q: Is the write protect on the disk covered? (y/n)
A: n
(Now R2 fails and R3 is selected).

Q: Remove all the resident programs if any.
Does the problem persist? (y/n)
A: y
(Now R3 fails and R4 is selected).

Q: Do you have a second disk or access to another compatible computer? (y/n)
A: y

Q: Does the disk work in the same way as in the other drive? (y/n)
A: y
Now R4 is confirmed and the diagnosis is "the bad floppy disk."

References

Addanki S, Cremonini R, Penberthy J S 1989 Reasoning about assumptions in graph of models. *Proc. Int. Joint Conf. Artif. Intell.* (Detroit, MI) 2: 1432–1438

Arun Kumar A T, Mahabala H N 1992 Multi-level qualitative reasoning applied to hydraulic circuits. Technical Report, Dept. of Comput. Sci. & Eng., Indian Inst. Technol., Madras

Collins J, Forbus K D 1987 Reasoning about fluids via molecular collections. *Proc. Am. Assoc. Artif. Intell. 87* (Seattle, WA) 1: 590–594

de Kleer J 1986a An assumption-based TMS. *Artif. Intell.* 28(2): 127–162

de Kleer J 1986b Problem solving with ATMS. *Artif. Intell.* 28(2): 197–224

de Kleer J, Brown J S 1984 A qualitative physics based on confluences. *Artif. Intell.* 24(1–3): 7–83

de Kleer J, Williams B C 1987 Diagnosing multiple faults. *Artif. Intell.* 32(1): 97–130

Dressler O, Farquhar A 1990 Putting the problem solver back in the driver's seat: Contextual control of the ATMS. *Lecture notes in artificial intelligence* (eds) J P Martin, M Reinfrank (Springer-Verlag) 515: 1–16

Falkenhainer B, Forbus K D 1988 Setting up large scale qualitative models. *Proc. Am. Assoc. Artif. Intell. 88* (St. Paul, MN) 1: 301–306

Forbus K D 1984 Qualitative process theory. *Artif. Intell.* 24(1–3): 85–168

Fukul C, Kawakami J 1986 An expert system for fault section estimation using information from protective relays and circuit breakers. *IEEE Trans. Power Delivery.* PRWD-1(4): 83–89

Hibler D L, Biswas G 1989 TEPS: The thought experiment approach to qualitative physics, *Proc. Int. Joint Conf. Artif. Intell. 89* (Detroit, MI) 2: 1279–1284

Kuipers B 1986 Qualitative simulation. *Artif. Intell.* 29(3): 289–338

Kurup R R, Mahabala H N 1992 Implementation of a constraint system. *Proc. Int. Assoc. Sci. Technol. Dev., Int. Symp. Modeling, Identification and Control,* Innsbruck Austria

Kurup R R, Mahabala H N 1993 Model-based diagnosis of power system network faults. Technical Report, Knowledge-Based Computer Systems Laboratory, Indian Inst. Technol., Madras

Liu Z Y, Farley A M 1990 Shifting ontological perspectives in reasoning about physical systems. *Proc. Am. Assoc. Artif. Intell. 90* (Boston, MA) 1:395–400

Mahabala H N, Kurup R R 1991a Implementation of an extended ATMS. Technical Report, Knowledge-Based Computer Systems Laboratory, Indian Inst. Technol., Madras

Mahabala H N, Kurup R R 1991b An implementation of a general diagnostic engine. *Proc. Int. AMSE Conference on Signals, Data & Systems* (Delhi: AMSE Press) 2: 137–150

Mahabala H N, Ravikanth G 1990 IITMRULE user's manual, KBCS Laboratory, Indian Inst. Technol., Madras

Mahabala H N, Ravi Prakash G, Managoli V V 1992 IITMDESS: A fault-tree based diagnosis shell, Technical Report, KBCS Laboratory, Indian Inst. Technol., Madras

Raiman O 1986 Order of magnitude reasoning. *Proc. Am. Assoc. Artif. Intell. 86* (Philadelphia, PA) 1: 100–104

Struss P 1988 A framework for model-based diagnosis. Siemens Report TR INF2 ARM-10–*88*, Munich

Talukdar S N, Cardozo E, Perry T 1986 The operator's assistant – an intelligent expandable program for power system trouble analysis. *IEEE Trans. Power Syst.* PRWS-1(3): 182–187

Vesonder G T, Salvatore J S, Zieliski J E, Miller F D, Copp D H 1983 ACE: An expert system for telephone cable maintenance. *Proc. Int. Joint Conf. Artif. Intell. 83* Karlsruhe, Germany 2: 116–121

Williams T L, Orgren P J, Smith C L 1983 Diagnosis of multiple faults in a nation-wide communications network. *Proc. Int. Joint Conf. Artif. Intell. 83*, Karlsruhe, Germany 2: 179–181