J. Indian Inst. Sci., May–June 1990, **70**, 269–281.

Diagnosis of steady-state process behavior using rules derived from signed directed graph

SAUGATA PRAMANIK* AND P. VENKATARAM
Electrical Communication Engineering Department, Indian Institute of Science, Bangalore 560 012, India.

Received on January 12, 1989; Revised on September 22, 1989.

Abstract

In this paper, we propose diagnosis of steady-state process behavior using rules derived from the signed directed graph (SDG) representing the interaction among the process variables. All abnormalities are correlated using backward reasoning through rules to yield a diagnosis. These rules can be added with experiential rules which are obtained from experts on plant operation.

Key words: Process behavior, fault diagnosis, hybrid knowledge, causal reasoning, SDG.

1. Introduction

In recent years, application of artificial intelligence (AI) to process industries using standard computer technology has emerged due to the growing complexity of modern process engineering, time constraints and limited availability of human expertise. One of the branches of AI increasingly becoming common in industries is knowledge-based systems (KBSs). Process plants *viz.*, power, chemical or petrochemical plants, etc., can be viewed as a domain which is knowledge-rich in terms of human expertise and in which a significant portion of the problem-solving is non-numeric. Knowledge-based approaches have an edge over the conventional methods as they employ efficient ways to capture the problem-solving knowledge from the domain-experts, explain the line of reasoning to the user, support modification and refinement of process knowledge, and capture and retain expertise that has accumulated many years of experience.

An important application of KBSs to have emerged in recent times relates to various types of diagnosis and troubleshooting. Most of the conventional knowledge-based fault diagnostic systems are based on expert knowledge compiled in the form of production rules. Parsaye and Lin[1] suggest building a knowledge base with rules derived from fault trees for emergency feed-water systems for nuclear power plants. Randhawa *et al*[2] propose an expert

*Present address: Knowledge-Based Computer System Development (KBCSD) Project, Super Computer Education and Research Centre, Indian Institute of Science, Bangalore 560 012.

system aid to troubleshooting of wave-soldering process using compiled rules. Chester *et al*[3] discuss an expert system approach to on-line fault diagnosis in chemical engineering domain.

The primary difficulty with most of these expert systems has been their lack of flexibility *i.e.*, they consist of rules which associate symptoms with a set of constraints specific to the process, and thereby limit the general applicability of the system. Moreover, these systems are unreliable in new situations as they rely on experience-based knowledge alone and fail when faced with unanticipated faults.

Another category of KBSs is based on a 'deep' model. By 'deep' we mean that the knowledge is basic knowledge concerning how and why a system works the way it does. This knowledge thus provides cause-and-effect information for troubleshooting a malfunction. The major goal of these models is to develop a fault diagnostic paradigm based on structure and behavior[4] of the system. Genesereth[5] discusses the use of design description based on structure and behavior for automated diagnosis. Narayanan and Viswanadham[6] present a methodology that merges graph and fault-tree-based failure analysis with rule-oriented reasoning. Rich and Venkatasubramanian[7] discuss a model-based approach for diagnosing prototype chemical plants.

The advantage of reasoning from a 'deep' model is that unanticipated situations may be explained to the extent of diagnostic resolution depending on the depth of process knowledge incorporated in the system. But 'deep'-level reasoning is often slow and tends to make the overall diagnosis computationally inefficient. Further in some ill-structured domains it is difficult to obtain the exact model of the process.

Attempts have been made to merge the experience-based and 'deep'-level approaches. The central idea is that whenever experience-based knowledge is unavailable or inadequate, 'deep' knowledge is utilized for diagnosis. Fink and Lusth[8] discuss an integrated diagnostic expert system in electrical and mechanical domain. Gallanti *et al*[9] describe an on-line process monitoring system which integrates experiential and 'deep' knowledge for this purpose. The only limitation of these systems is that they lack flexibility in their diagnosis and are unable to adapt to other types of process plant configurations.

We propose a hybrid knowledge framework[10,11] which includes a process-independent diagnostic mechanism based on causal and qualitative reasoning, and integrates knowledge based on experience and 'deep' knowledge about structure and behavior of the process plant. In this paper, we present only the diagnosis of steady-state process behavior which comprises the second stage of our research project[12]. This diagnosis technique is based on formation of rules derived from the process digraph which represents the behavioral knowledge. Before discussing the approach in detail we outline some of the salient features of the diagnostic system as a whole.

1.1. *Outline of the hybrid knowledge-based fault diagnostic system*

In the system proposed, we consider a process plant as an assembly of several functional subsystems or functional blocks (FBs), which operate to meet an identifiable goal. This partitioning reduces the problem of searching a malfunction cause to a manageable size. Each FB consists of a hybrid knowledge required to guide the diagnosis rapidly towards a solution. The knowledge is hybrid because it involves various types:

(a) 'Shallow' or experiential knowledge,
(b) 'Deep' knowledge: (i) structural and (ii) behavioral.

Shallow' or experiential knowledge is experience-oriented knowledge that captures empirical associations and expertise that is gained through repeatedly diagnosing a problem.

'Deep' knowledge includes knowledge concerning how and why a system works in the way it does, thus providing cause-and-effect information needed when diagnosing a malfunction. The structural knowledge captures the physical layout of the plant *i.e.*, component's part, connectivity and associated instrumentation. The behavioral knowledge comprises knowledge about causal interactions among process-dependent parameters (variables) and their states.

In addition to the hybrid knowledge, control knowledge is incorporated in the system's knowledge base. This is a set of instructions on how different pieces of process knowledge should be accessed and processed for diagnosis. The diagnostic system also includes a plant database which consists of data obtained from alarms, sensors or instrumentation. These data are represented in the form of facts and may be updated through a data-acquisition system.

The diagnostic mechanism (DM) of the system is composed of three consecutive phases for locating a fault (fig. 1). The first phase is malfunction block identification (MBI) which locates a malfunctioning FB or malfunction block (MB), based on alarm data whenever violation of process parameters occurs. Once the suspected MB is identified, the second phase *viz.*, malfunction parameter identification (MPI) is invoked to locate parameters which indicate the prime cause(s) of the fault in that MB. Finally, malfunction component identification (MCI) phase is invoked to locate the malfunctioning component based on the results supplied by MPI. The DM is process independent and is capable of adapting to various types of plant configurations.

As mentioned earlier, presentation of the MPI phase, which diagnoses the steady-state behavior of the process, is the main objective of this paper. We first discuss the behavioral knowledge organization and then focus on the MPI phase of DM.

2. Behavioral knowledge organization

Behavior may be described as a relationship between the input and output of a system or a component. Genesereth[5] defines behavior as equations, rules, procedures that relate a system's input, output and state. In effect the behavioral knowledge of a process comprises knowledge about causal interactions among process-dependent parameters and their states.

In a process plant environment, a large fraction of behavioral knowledge is incorporated

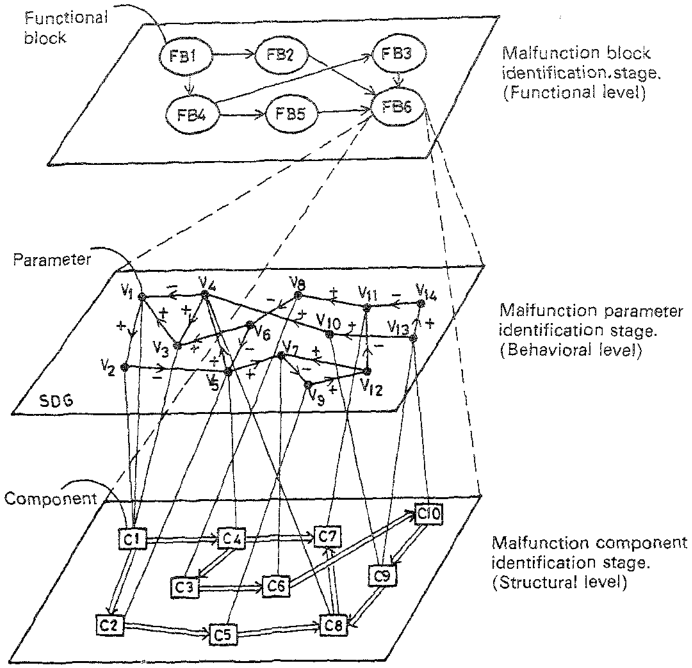
Functional block
FB1
FB2
FB3
FB4
FB5
FB6
Malfunction block identification.stage. (Functional level)
Parameter
V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
SDG
Malfunction parameter identification stage. (Behavioral level)
Component
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
Malfunction component identification stage. (Structural level)

FIG. 1. Overview of the diagnostic mechanism.

in the physical model based on thermodynamics, heat, mass, and momentum transfer. We discuss this keeping in mind the steady-state behavior of the process. We adopt a qualitative* steady-state model for behavioral diagnosis to test our prototype diagnostic system.

2.1. *The qualitative representation*

The behavioral knowledge is represented as a signed directed graph (SDG)[15]. The vertices

*Obtaining an exact quantitative model for a process is difficult. Even if such a model is obtained, it is time consuming and cannot meet real-time constraints for on-line diagnosis[13]. By abandoning the precision of numerical information, a qualitative model gains the ability to reach conclusions even with a little information[14]. The disadvantage is that the predictions are often ambiguous; where traditional quantitative model gives an exact prediction, a qualitative model often narrows the possibilities to a small set. When it is not adequate, the framework provided by the qualitative model can guide one to apply more precise knowledge.

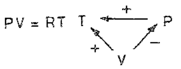

FIG. 2(a). The real gas-state equation and its SDG.

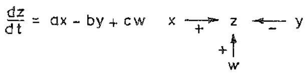

FIG. 2(b). A linear differential equation and its SDG.

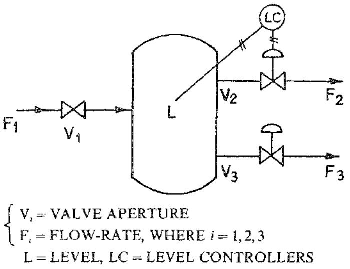
LC
L
F1
V1
V2
F2
V3
F3
V_i = VALVE APERTURE
F_i = FLOW-RATE, WHERE i = 1, 2, 3
L = LEVEL, LC = LEVEL CONTROLLERS

FIG. 2(c). A water tank system (reproduced from Iri[15] *et al*).

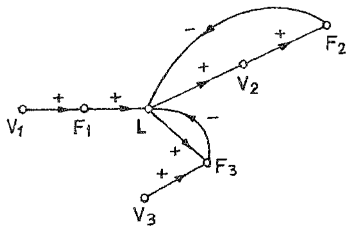

FIG. 2(d). The SDG of the water tank system of fig. 2(c). (reproduced from Iri[15] *et al*).

of the SDG correspond to process parameters (variables) and the edges correspond to the cause-effect relationship between process parameters. The direction of deviation of the process parameters is represented by sign on the edges: $+(-)$ indicates the tendency of cause and effect vertices to change in the same (opposite) direction. Each vertex is assigned a three-level qualitative value: $\{-1, 0, 1\}$. A vertex is assigned 0 if it is within a specified tolerance range, -1 if it is below, and 1 if it is above the specified tolerance range. To illustrate the SDG representation, consider the following instances:

i) The equation of state for a unit mass of a real gas is given by $PV = RT$, where P, V, T, and R correspond to pressure, volume, temperature and universal gas constant, respectively. The SDG for this equaion is shown in fig. 2(a).

ii) A linear differential equation and its corresponding SDG is shown in fig. 2(b).

iii) For the water tank system of fig. 2(c) the influence among the parameters is represented by the SDG of fig. 2(d).

The motivations behind adopting the SDG representation are as follows. Firstly, it is easy to create the SDG either from design equations or from the knowledge of physical mechanism of a process in qualitative term or both. Secondly, the SDG representation is suitable for causal reasoning that we use for the purpose of diagnosis. Finally, with the help of SDG the dynamic behavior of a process may be simulated to check its performance.

2.2. *Acquisition of the SDG*

The SDG may be derived from the design equations (quantitative model), confluence[16] equations (qualitative differential equations) and constraints[17] from material and energy balances. Usually the SDG is created manually. It requires only qualitative and subjective judgement of a person, generally a process engineer, to transform the physical model into an

SDG. Automated SDG generation directly from the design equations, confluence equations and constraints is an important issue for further research.

2.3. *Representation of the SDG in the form of rules*

The SDG representation is only an intermediate one. The SDG is finally converted into a set of rules along with some control premises required for the purpose of diagnostic reasoning.

The primary proposition of SDG-based techniques is that cause and effect linkages must connect the fault origin to the observed symptoms of the fault. Each edge in the SDG represents a fundamental unit of interaction. For instance, a 'negative' edge $A \text{—}(-)\rightarrow B$ portrays that if the state of A becomes high (low), then the state of B becomes low (high). For a 'positive' edge the effect is reversed. A acts as the antecedent and B as the consequent. In other words, if A becomes 1 (-1), then the state of B becomes -1 (1), for a 'negative' edge. However, irrespective of the sign on the edge, if state of A is 0 then the state of B is 0.

For simplicity, we note the following assumptions:

(1) The failure propagation time between any two adjacent vertices is zero. (The failure propagation time would be crucial if we had considered component digraph where process units are represented as vertices.)
(2) The failure propagation probability between any two adjacent vertices of the SDG is 1. (For the present discussion we assume that the interaction of process parameters is approximately definite and uniform.)
(3) One and only one antecedent at any point of time can affect the consequent. (This assumption is justified if no abnormalities are simultaneously occurring and if each parameter undergoes only one transition between qualitative states during fault propagation.
(4) Each parameter undergoes no more than one transition between qualitative states during fault propagation.

If we examine the SDG for the linear differential equation given in fig. 2(b), using the above assumptions the following rule may be applied to describe the antecedent-consequent behavior:

$$\text{IF } X \langle\rangle 0 \text{ THEN } Z = X$$

OR

$$\text{IF} - Y \langle\rangle 0 \text{ THEN } Z = -Y$$

OR

$$\text{IF } W \langle\rangle 0 \text{ THEN } Z = W.$$

In other words, Z is assigned the value of X, or $-Y$ or W whichever is violated, assuming a normal (0) steady-state value initially. Z is normal (0) when X, Y and W are all normal. This rule may be represented in PROLOG as follows:

```
var(z, Z):-var(x, X), X <> 0, Z = X;
           var(y, Y), Y <> 0, Z = - Y;
```

```
            var(w, W), W<>0, Z = W.
     var(z,0).
```

The clause var(p, P) represents the state P of a parameter 'p'.

Since rules are efficient at causal reasoning (backward, forward or both), it is natural to convert the SDG into a set of rules as stated above. In this approach, we further extend the condition (IF-part) by adding certain control premises whose inclusion expedites the diagnostic reasoning. We designate this modified version of the above rules as SDG-rules. The SDG-rules, corresponding to the SDG of fig. 2(d), are as follows:

```
var(f1, X):-dat(f1,X), X<>0, tl(f1):
        test(f1), loop(v1), var(v1,X).
var(f1,0):-el(f1).
var(v1,X):-dat(v1,X), X<>0, tl(v1).
var(v1,0):-el(v1).
var(v3,X):-dat(v3,X), X<>0, tl(v3)
var(v3,0):-el(v3).
var(l,X):-dat(l,X), X<>0, tl(l);
      test(l), loop(f1), var(f1,X);
      test(l), loop(f2), var(f2, Y), X = -Y;
      test(l), loop(f3), var(f3, Y), X = -Y.
var(l,0):-el(l).
var(f2,X):-dat(f2,X), X<>0, tl(f2);
        test(f2), loop(v2), var(v2, X).
var(f2,0):-el(f2).
var(f3,X):-dat(f3,X), X<>0, tl(f3);
        test(f3), loop(v3), var(v3,X);
        test(f3), loop(l), var(l,X).
var(f3,0):-el(f3).
var(v2,X):-loop(l), var(l, Y), X = -Y.
```

We have assumed that the parameters $f1$, $f2$, $f3$ and l are measured. The clause dat(p, X) retrieves the state X of the measured parameter p from the process database. The clause loop(p) takes care of loops or feedbacks incident on the vertex p. The remaining clauses $tl(p)$, $el(p)$ and test(p) will be explained later.

The SDG-rules are also used to explain the fault cause once a fault is located[12]. Finall experience-oriented rules may be added to the SDG-rules to speed up the diagnosis. W may have a rule as follows:

IF parameters $p1$ AND $p2$ AND $p3$ AND... are violated

THEN components $c1$ OR $c2$ OR $c3$ OR... are faulty.

For instance,

IF controller 1 output is low (*i.e.*, CNT 1 = -1)

AND valve 2 position is high (*i.e.*, VLV 2 = 1)

THEN valve 2 is probably faulty.

The SDG-rules are automatically generated by knowledge-acquisition interface (KAI) which takes the SDG as an input and converts it into a set of SDG-rules. KAI is also useful for adding, deleting or modifying the SDG and the associated SDG-rules.

3. Causal reasoning

A causal model usually portrays what is happening in a system, what caused to happen and what will happen in the system. Process plants provide various paths of interaction which show the existence of a theory of causality. The causal model we consider consists of the behavioral knowledge in the form of SDG-rules derived from the process SDG that contains both measured and unmeausured vertices (parameters).

We presume that the MB which is suspected to be responsible for process malfunctioning is identified[12]. The second phase *viz.*, MPI is invoked to locate the parameters that indicate the prime cause(s) of the fault in the MB. The MPI algorithm is based on causal and qualitative reasoning and is given in the following section.

3.1. *The MPI algorithm*

In this algorithm, the SDG-rules which represent the interaction among various process parameters are evaluated using process database. All abnormal parameters are correlated using backward reasoning through these rules to yield diagnosis.

Problem definition: Given SDG-rules pertaining to an MB, and the qualitative states of all measured parameters, the problem is to locate a set of violated parameters which are causally 'independent' (we designate such parameters as malfunction parameters; they indicate the prime cause(s) of malfunction in the MB).

{*Comment*: The algorithm is based on graph traversal. The input is a set of measured parameters of a suspected MB. The output is a set of parameters which are not causally dependent on any other parameter in the MB, and holds responsibility for the declared effect.}

Method: The MPI algorithm is composed of the following steps:

Step 1: Gather all abnormal measured parameters (AMPs) reported by process database as an unexplored list, UL = {AMP_1, AMP_2, $AMP_3, \ldots,$ AMP_j} where $j \leqslant k$, k is the total number of measured parameters in the MB.

Step 2: Select the first element of UL (call it active element (AE)) and apply modified depth first search (MDFS) algorithm to find out all AMPs which are causally related to AE, and store them in a temporary list (TL), TL $\subseteq$ UL. Delete AE from UL after the current MDFS terminates.

Step 3: If (TL $= \phi$) OR ($AMP_i \in$TL) AND ($AMP_i \notin$UL)) where $i = 1, 2, \ldots j$ then the contents of UL are unchanged; else all the elements of TL are added in front of UL without duplication.

Step 4: Store AE in an explored list (EL), iff TL $= \phi$ at the end of current MDFS.

Step 5: Continue steps 2 through 5, till UL $= \phi$. If UL $= \phi$ and EL $= \phi$, then add AE to EL and stop.

The MDFS is a modified version of the depth first search (DFS) algorithm in which the DFS is carried out through the SDG-rules with those parameters in the consequent (THEN-part) that are antecedents of AE. The following cases may occur:

Case 1: Whenever an SDG-rule with a normal measured parameter as its consequent is encountered, the DFS is terminated through that rule.

Case 2: Whenever an SDG-rule with an AMP as its consequent is encountered, the DFS is terminated through that rule, and the AMP is stored in TL.

Case 3: Whenever an SDG-rule with any parameter as its consequent is encountered more than once, the DFS is terminated through that rule, thereby infinite causal looping is avoided.

Limitations: The MPI algorithm correlates different AMPS through causal linkages (described by antecedents and consequent in the SDG-rules) to converge the search towards the cause of the fault. The MPI algorithm terminates when UL is null, creating the EL which contains a set of causally independent AMPs. The AMPs that are present in EL indicate the possible prime causes of the fault in the MB and are thus the malfunction parameters. However, it is impossible to further reduce the number of these parameters using the SDG information. Moreover, they are ambiguous with regard to fault sources due to single and multiple origins.

Finally, the selection of a dominant causal antecedent from a set of potential antecedents is arbitrarily fixed in the SDG-rules. No criteria has yet been adopted for optimally ordering the antecedents in the SDG-rules.

3.2. *Example*

Let us consider the buffer tank system of fig. 3(a) and its corresponding SDG given in fig. 3(b). The SDG-rules for this system are as follows:

```
var(f1, X):-dat(f1, X), X < > 0, tl(f1);
        test(f1), loop(v1), var(v1, X).
```

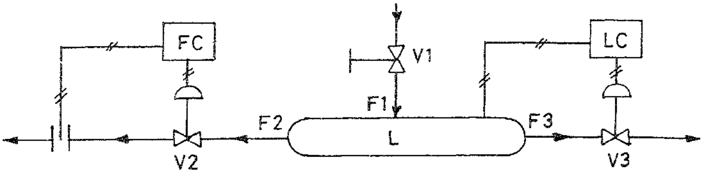

FIG. 3(a) The buffer tank system (reproduced from Viswanadham[20] *et al*).

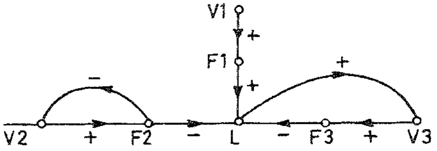

FIG 3(b). The SDG of the buffer tank system of fig. 3(a) (reproduced from Viswanadham[20] *et al*).

```
var(f1, 0):-el(f1).
var(v1, X):-dat(v1, X), X <> 0, tl(v1);
var(v1, 0):-el(v1).
var(l, X):-dat(l, X), X <> 0, tl(1);
            test(l), loop(f1), var(f1, X);
            test(l), loop(f2), var(f2, Y), X = − Y;
            test(l), loop(f3), var(f3, Y), X = − Y.
var(l, 0):-el(l).
var(f2, X):-dat(f2, X), X <> 0, tl(f2);
              test(f2), loop(v2), var(v2, X).
var(f2, 0):-el(f2).
var(f3, X):-dat(f3, X), X <> 0, tl(f3);
              test(f3), loop(v3), var(v3, X).
var(f3, 0):-el(f3).
var(v2, X):-loop(f2), var(f2, Y), X = − Y.
var(v3, X):-loop(l), var(l, X).
```

The clause *tl*(*p*) appends a parameter *p* to the list TL. The clause *el*(*p*) appends *p* to the list EL. The clause test(*p*) ensures that the MDFS is started from the antecedent of AE only.

Table I
Status of measured parameters

V1	F1	L	V2	F2	V3	F3
	0	+		—		+

Table II
Status of different lists at different execution stages

Execution stages	Unexplored list (UL)	Temporary list (TL)	Explored list (EL)
0	[L, F2, F3]	[]	[]
1	[F2, F3]	[F3, F2]	[]
2	[F3]	[]	[F2]
3	[]	[L]	[F2]

Table I depicts the status pattern of some measured parameters at an instant of time. Table II shows the status of UL, TL, and EL at different stages of MPI execution. The MDFS starts from the SDG-rule with l as consequent and obtains $f2$ and $f3$ as elements of TL. The UL is subsequently modified and the MDFS is invoked from the SDG-rule with $f2$ as consequent in the second pass. As $f2$ depends on $v2$ and *vice versa*, a loop is recognized. Since $f2$ does not depend on any other parameter except $v2$, TL becomes empty and thus, $f2$ is stored in EL. Finally, the MDFS is invoked from the SDG-rule with $f3$ as consequent resulting in l in TL and an empty UL. Since UL is empty, MPI terminates. Because $f2$ is an element of EL, it is identified as malfunction parameter that indicates the fault in buffer tank system. Although $v2$ is a part of the causal loop formed by $f2$, it is not selected as it is an unmeasured parameter.

4. Implementation

The initial task *i.e.*, acquisition of the domain's hybrid knowledge is done with the help of KAI, which is an interface program implemented in TURBO PASCAL[18]. However, all the phases of DM are implemented with TURBO PROLOG[19]. We adopted PROLOG for conceptualizing, creating and prototyping the diagnostic system. It is an easy task to implement KBSs using PROLOG directly for knowledge representation and inference mechanism. Conciseness of PROLOG programs with resulting decrease in development time makes it an ideal language for prototyping small to medium-sized KBSs.

The overall implementation of the prototype diagnostic system has been carried out on an IMB PC-XT which supports both TURBO PASCAL and TURBO PROLOG programming environment.

5. Conclusions

The method of diagnosis of steady-state process behavior presented in this paper is simple and direct. If the limitations are acceptable, the diagnostic reasoning is straightforward. However, efforts are being pursued to overcome these limitations, the accomplishment, of which will be significant.

In this work we tried to establish that the rules are efficient for causal reasoning. Rules, in general, facilitate reasoning forward, backward or both. The SDG-rules are also used to explain the fault-cause once a fault is located. Finally, the rule representation assists the knowledge engineer to add rules obtained from the human experts who have acquired the diagnostic knowledge about the behavior of the process in a compiled form, developed it through experience or derived it from a 'deeper' model. The integration of SDG-rules with experiential knowledge increases the diagnostic efficiency during common faults and makes the system capable of tackling unanticipated faults when the latter fails.

Acknowledgment

The authors are greatly indebted to Prof. V. Rajaraman for a series of discussions in this research.

References

1. PARSAYE, K. AND LIN, K. Y. An expert system structure for automatic fault tree generation for emergency feedwater systems for nuclear power plants, *IEEE Western Conf on Expert Systems*, 1987, pp. 25–30.
2. RANDHAWA, S. U., BARTON, W. J. AND FARUQUI, S. Wavesolder assistant. An expert system aid to troubleshooting of the wave soldering process, *Computers Ind. Engng*, 1986, **10**, 325–334.
3. CHESTER, D., LAMB, D. AND DHURJATI, P Rule-based computer alarm analysis in chemical process plants, *Proc. 7th. Micro Delcon*, 1984, pp. 22–29, IEEE Computer Society Press.
4. DAVIS, R Diagnostic reasoning based on structure and behavior, *Artif. Intell.*, 1984, **24**, 347–410.
5. GENESERETH, M. R. The use of design description in automated diagnosis, *Artif. Intell.*, 1984, **24**, 411–436.
6. NARAYANAN, N. H. AND VISWANADHAM, N A methodology for knowledge acquistion and reasoning in failure analysis of systems, *IEEE Trans.*, 1987, **SMC-17**, 274–288.
7. RICH, S. H. AND VENKATASUBRAMANIAN, V. Model-based reasoning in diagnostic expert system for chemical process plants, *Computers Chem. Engng*, 1987, **11**, 111–122.
8. FINK, P K. AND LUSTH, J. C. Expert systems and diagnostic expertise in the mechanical and electrical domains, *IEEE Trans.*, 1987, **SMC-17**, 340–349.
9. GALLANTI, M., GILARDONI, L., GUIDA, G., STEFFANI, A. AND TOMADO, L. Integrating shallow and deep knowledge in the design of on-line process monitoring system, *Int. J. Man–Mach. Stud.*, 1987, **27**, 641–664.

10 PRAMANIK, S AND VENKATARAM, P. A hybrid knowledge structure for process plant fault diagnosis, *Proc. Parallel Processing Systems and their Applications, 4th. Natn Conv of Computer Engrs*, The Institution of Engineers (India), Calcutta, Dec. 1987, pp. 124-129.

11. PRAMANIK, S. AND VENKATARAM, P. A hybrid knowledge-based fault diagnostic system for chemical process plants, *Proc 1st Natn Conf. on Knowledge-Based Computer Systems*, CSI, Bangalore Chapter, June 1988, pp. 18–21.

12. PRAMANIK, S A hybrid knowledge-based system for process plant fault diagnosis, M.Sc. (Engng) Thesis, Deptt of Electrical Communication Engng, Indian Institute of Science, Bangalore, 1989.

13. ISHIDA, Y. An application of qualitative reasoning to process diagnosis: Automatic rule generation by qualitative simulation, *IEEE Conf on Artif. Intell. Applic.*, 1988, pp. 124–129.

14 MILNE, R. Strategies for diagnosis, *IEEE Trans*, 1987, **SMC-17**, 333–339.

15. IRI, M., AOKI, K, O'SHIMA, E. AND MATSUYAMA, H. A graphical approach to the problem of locating the origin of system failure, *J. Operations Res. Soc. Jap.*, 1980, **23**, 295–310

16. KLEER, J. D. AND BROWN, J. S. A qualitative physics based on confluences, *Artif Intell.*, 1984, **24**, 7–83.

17 LEARY, J. J. Process fault detection using constraint suspension, *Proc. IEE*, 1987, **134**, 264–271.

18. *Turbo Pascal Reference Manual*, Borland International Inc, Scotts Valley, CA, 1986.

19. *Tubro Prolog User's Guide*, Borland International Inc, Scotts Valley, CA, 1986.

20. VISWANADHAM, N., SARMA, V. V. S. AND SINGH, M. G. *Reliability in computer and control systems*, 1987, North-Holland.