ICOT Technical Memorandum: TM-0603

TM-0603

A Human Stractegy-Based Troubleshooting Expert System for Switching Systems

by
S. Wada, Y. Koseki (NEC) & T. Nishida

March, 1989

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1 Chome
Minato-ku Tokyo 108 Japan
(03) 456-3191 ~ 5
Telex ICOT J32964

Institute for New Generation Computer Technology

A HUMAN STRATEGY-BASED TROUBLESHOOTING EXPERT SYSTEM FOR SWITCHING SYSTEMS

Shin-ichi Wada, Yoshiyuki Koseki and *Tetsurou Nishida*

C&C Systems Research Laboratories, Integrated Switching Development Division

NEC Corporation
4-1-1 Miyazaki, Miyamae-ku, Kawasaki, 213 Japan

ABSTRACT

This paper presents a human strategy-based diagnostic method in a troubleshooting expert system for electronic switching systems. The system infers defective components based on an *abstract-signal-flow* model, which makes it possible to describe easily the complex structure of the target switching system. Several kinds of knowledge of symptoms, tests and equipment structure are concisely represented in a uniform network style. In addition to these features, truth maintenance facilities are incorporated for indeterminate inferences. Inferences which utilize as many default assumptions as possible and belief revision due to later discoveries of contradictions are properly realized. The facilities also enable flexible diagnosis, such as reasoning based on anticipated test results without actually executing the tests. The method for environment control enables this kind of reasoning. The paper summarizes the architecture and features of the troubleshooting expert system and presents a diagnostic method based on truth maintenance.

1 INTRODUCTION

Modern electronic devices like an ESS (Electronic Switching System) are very complex and difficult to diagnose. Although diagnostic test functions are built into ESS's based on the hardware technology, defective components are not always clearly found by using them. By contrast, highly skilled human experts can locate defective components and repair them by using various diagnostic strategies. This has motivated development of a troubleshooting expert system modeled on human experts' strategies. Some of the features of its methods and architecture are as follows:

(1)The system utilizes several heuristic test methods of human experts in order to locate the defective components. It also analyses the symptoms in order to guess the defective components.

(2)The system has a kind of structural knowledge of the target ESS called an *abstract-signal-flow* model. The development and modification of the knowlege base in this style is far easier than that in diagnosis systems using empirical symptom-cause associating rules like MYCIN(Shortliffe 1976).

(3)Several kinds of knowledge are represented in a unified network by utilizing the object-oriented facilities in ESP (Extended Self-contained Prolog)(Chikayama et al. 1984). By the representation method, the knowledge base is kept compact and modifiable.

(4)The system is designed to allow much flexibility in man-machine interaction. Therefore, an inexperienced technician can perform diagnosis just by following the recommended actions, while an experienced technician can freely choose alternative actions based on his judgment.

A prototype system with these features was developed on a PSI-machine (Taki et al. 1984). Moreover, truth maintenance facilities(Doyle 1979; de Kleer 1986a; de Kleer 1986b) have been incorporated into the system in order to achieve indeterminate inferences.

In the inferences of suspected components, there are many cases where definite conclusions cannot be made because of exceptions. The inferences done by human experts are heuristic and therefore might be erroneous. In order to narrow down the suspected components, these kinds of indeterminate inferences should be allowed. In order to make indeterminate inferences and properly revise them when they are known as erroneous, truth maintenance is necessary.

In the diagnostic problem domain, reasoning methods based on truth maintenance techniques have recently been proposed(de Kleer and Williams 1987; Young 1987). They make assumptions which represent the functionalities or violations of the devices, and they do reasoning on the device behaviors based on these assumptions.

On the other hand, the expert system in this paper uses assumptions to represent the indeterminate factors included in diagnostic inferences, and it infers suspects based on them. Among many possibilities in selecting assumptions to use, the expert system adopts a control method which makes use of as many default assumptions as possible. By this control method, suspected com-

ponents are narrowed down as much as possible, even if some contradictions are discovered. In addition, the method enables reasoning based on anticipated test results without actually executing the tests. By truth maintenance and the control method, more flexible and effective diagnosis is achieved.

The rest of this paper consists of two parts. The first part presents the objectives of the troubleshooting expert system, its architecture and features. The second part describes the incorporation of truth maintenance including motivation, the diagnosis method based on truth maintenance, and the environment control method.

2 OBJECTIVES OF THE DEVELOPMENT

2.1 Problems in ESS Diagnosis

Modern ESS's are very complex and difficult to maintain. When a trouble occurs in an ESS, one or more fault messages are printed out on an ESS console. In some cases, a trouble is known from telephone subscriber's complaints. However, it is difficult for inexperienced maintenance personnel to pinpoint the cause of trouble from these kinds of information.

In order to locate the defective component, an ESS has built-in diagnostic test functions. However, the performance of these functions is limited, and a defective component cannot always be clearly detected by them.

2.2 Diagnosis by Human Experts

Contrary to the limits of built-in diagnostic functions, a highly-skilled maintenance technician can find the defective component and repair it. He diagnoses by various strategies based on his knowledge and skills. For example, he analyzes the symptom in detail and guesses the suspected components. In order to determine a cause among multiple suspects, a human expert thinks of effective tests to narrow them down. The word *test* means any action to reduce the number of suspects. Some of the tests carried out by human experts are as follows:

- Viewing the occurrences of symptoms and their distribution
- Acquisition of various kinds of information in the ESS, such as acquisition of operation status in duplicated units
- Typing commands to the ESS and watching the resulting ESS behavior
- Changing the operation status for the duplicated units and looking for trouble occurrences

After narrowing down the suspects, he tries repair actions such as replacing suspect FRU's(Field Replaceable Units) with spares until the symptom disappears.

2.3 Necessity of an Expert System Modeled on Human Experts

As described above, highly skilled technicians carry out diagnosis. However, there are few technicians who have the ability to do so. In addition, since the life span of an ESS is over ten years, it is hard to retain highly skilled maintenance personnel in a specific telephone office for a long time.

Therefore, an expert system is desired which assists inexperienced maintenance technicians in performing diagnosis of an ESS based on the strategy used by human experts.

3 BASIC ARCHITECTURE

3.1 Basic Diagnosis Flow

The basic diagnosis flow in the expert system is shown in Fig-1. During the diagnosis, *suspected components*, that is, candidates which might have caused the trouble, are considered and updated.

Given the symptom information, the expert system first considers the suspected components by analyzing the symptom information. After that, a test effective for them is chosen and carried out. By interpreting the test result, suspected components are narrowed down; the list of suspected components is updated to smaller ones. According to the updated list of suspected components, effective tests are reconsidered. In this way tests and their interpretations are repeated. After several repetitions, a small number of suspected components can be obtained. Finally, corresponding FRU's, such as equipment packages or connector cables, are replaced, and the trouble is repaired.

3.2 Abstract-signal-flow Based Reasoning

In order to implement reasoning about suspected components according to the symptom and test results,

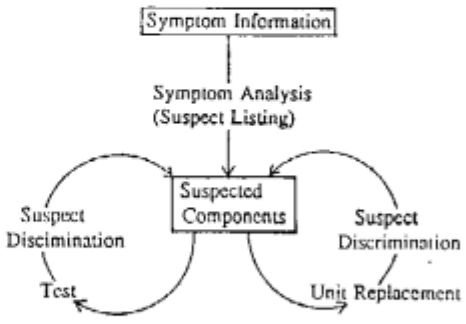


Fig-1: Basic Diagnosis Flow

the expert system has an *abstract-signal-flow* description, which represents the functional structure of the ESS. The expert system looks at this description when considering suspected components. The description is represented as a directed graph. Fig-2 shows an extraction of the abstract-signal-flow description. Each node in it represents a functional component, and each arc represents a signal-flow between components.

With an abstract-signal-flow description, suspected components for a given symptom can be enumerated. For example, consider a case wherein the following symptom is occurring:

SYMPTOM: no-dial-tone
"Although a subscriber picks up a receiver, no dial-tone is heard."

In order for a dial tone to be normally heard, the following signal path must not be defective:

SIGNAL PATH:
"a control signal path from the line to the CPU"

The components on the path can be listed by searching the abstract-signal-flow description. By enumerating the nodes along the path, from *Line* to *CPU* through *ctl_up* arcs in Fig-2, the suspected components for the no-dial-tone symptom are listed:

SUSPECTED COMPONENTS:
Line, LC, E/G Conv, LM Intf, Scn Ctl, SMUX, ... , CPU

The abstract-signal-flow method is also effective in narrowing the suspected components based on test results such as those of signal-flow checking tests.

Structural knowledge of the ESS for the abstract-signal-flow method is represented in suitable precision. In addition, the total size of the knowledge base may be kept small. This is because a signal-flow description is shared by many kinds of symptoms. As a result of utilizing the abstract-signal-flow method, the design experts can easily present their knowledge.

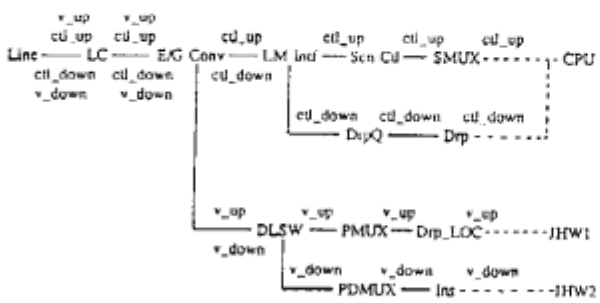


Fig-2: Abstract-signal-flow description

3.3 Network-based Knowledge Representation

The expert system is based on multiple kinds of knowledge acquired from both design experts and maintenance experts of the target ESS. Structural knowledge regarding the ESS is acquired from the design experts, while the pertinent maintenance knowledge, such as symptoms and tests specific to the ESS, is acquired from maintenance experts.

In the expert system these kinds of knowledge are represented in a unified network. Signal flows between functional blocks are directly represented as relations. Other relations between different kinds of knowledge, such as symptom, test, functional blocks and FRU's are also represented in the same style. Fig-3 shows an example of a network-based representation. In the figure, functional blocks corresponding to Symptom-A can be listed as follows:

Block-A, Block-B, and Block-C

In this way, several kinds of knowledge are compactly described in network-based representation. In addition, a local modification of the network can be more easily performed than equivalent modification in symptom-cause associating rules such as MYCIN.

The network for this representation is described by object-oriented facilities in ESP. Relations between objects are represented by slots in objects. In addition, symptom, test and structural knowledge is represented without redundancy in hierarchical structures. For instance, common properties among several symptoms can be described in a general symptom object by an *inheritance* facility.

3.4 Flexible Man-machine Interface

The expert system provides flexible man-machine interaction in selecting diagnostic actions. In selecting a subsequent action, such as a test execution or unit

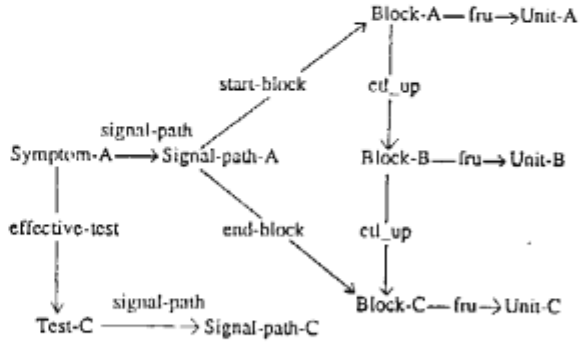


Fig-3: Network-based Knowledge Representation

replacement, the system reasons about possible candidates and recommends the most appropriate one, leaving the maintenance technicians flexibility to select an alternative action. Based on such flexible man-machine interaction, an inexperienced maintenance technician can carry out troubleshooting merely by following the actions recommended by the system, while an expert technician may select actions himself.

The system provides a facility of displaying the suspects narrowing based on hypothetical test results designated by the user. This helps the user understand the effectiveness of test execution. This is easily implemented by the truth maintenance facilities, which will be explained in 5.4.

In addition, the system provides a user friendly interface for easier operation, based on the multiple window environment of a PSI-machine. An example of the user-interface screen is shown in Fig-4.

4 INCORPORATION OF TRUTH MAINTENANCE

This section presents the motivation for incorporating truth maintenance facilities into the expert system. The inferences for narrowing down suspects by test results are frequently based on indeterminate assumptions or preconditions. Many kinds of reasoning proceeds from these assumptions. For instance in the narrowing down of suspects, there underlie the following assumptions:

- The fault is non-intermittent.
  (The fault status continues and docs not disappear.)
- The power is normally supplied to the ESS.
- The clock distribution functions normally.

There is another kind of assumptions which underlie indeterminate inferences. For instance, if fault symptoms are found in one area and not in another, then it can be supposed that the cause is limited to the specific area. However, this is uncertain because the absence of fault symptoms might be accidental. If the observation time were longer, faults might have also been found in another area.

These kinds of indeterminate inferences are performed by human experts. These are not confirmed strategies but are usually effective in narrowing down the suspects and quickly finishing the diagnosis.

However, these inferences sometimes lead to erroneous conclusions. An expert realizes his misjudgment when he rules out all suspects by replacing suspected physical units. In some cases, he realizes his error when he finds ESS behavior contradictory to his beliefs. When he discovers a contradiction, he changes his belief. He casts doubt on the assumptions producing the contradiction and abandons some of them. He then believes a new consistent set of assumptions.

In order to realize these kinds of indeterminate inferences of human experts, truth maintenance facilities like TMS(or ATMS) are incorporated into the expert system. TMS manages the dependency relations among facts, assumptions and conclusions. Based on this, TMS enables the maintenance of consistency between them.

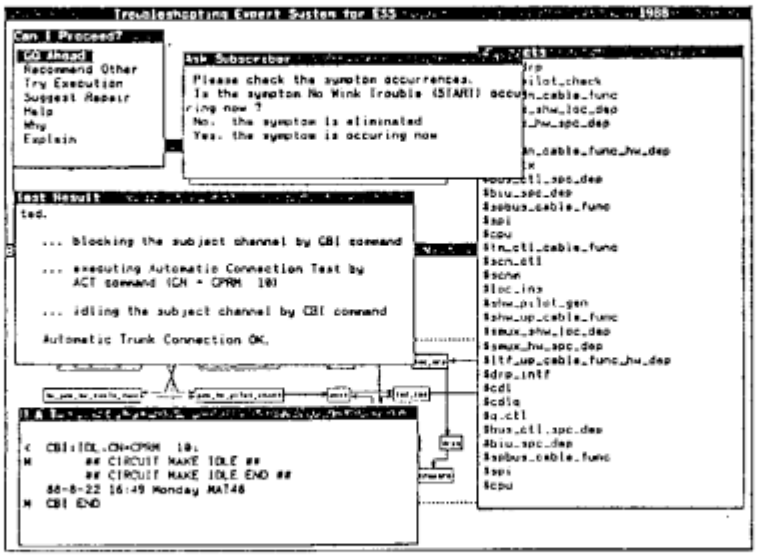


Fig-4: User interface screen

There is another motivation for incorporating truth maintenance. It is desired to infer suspects based on anticipated test results without actually executing the tests. That is, there are some tests whose results are anticipated, and it is desired to omit the execution and proceed to narrow down the suspects based on the anticipated results. This strategy is not always reliable, but it is effective for quick diagnosis. In addition, the strategy is desirable for the kind of tests which are temporarily detrimental to the service quality.

For example, when an alarm lamp is not on, there is the possibility of executing a test to check the condition of the alarm lamp itself. The anticipated result is that the alarm lamp is normal. From this result, the following deduction becomes possible, and the suspects will be narrowed down:

> *"If an alarm is not lighting, and the alarm lamp itself is normal, then suspects are limited to ....."*

In this situation, it is faster to assume the normality of alarm lamp than to check its condition. In this way, omission of the test is desired.

However, when it is suspected that assumptions cause a contradiction, these omitted tests must be performed. By utilizing the truth maintenance facilities, this kind of test omission control can be realized.

In the domain of diagnostic problems, de Kleer and Williams (1987) present a method of diagnosis based on ATMS. However, their method differs from our approach. Their method introduces assumptions in order to represent the destruction of a target system and enables diagnosing multiple faults. Our approach, on the other hand, utilizes assumptions to represent the indeterminate factors in the inferences of suspected components. Their method does not consider this kind of indeterminateness.

Young (1987) also utilizes default reasoning in troubleshooting. In his method, each belief represents a fault candidate, and a belief revision is a change of the fault candidates. Based on this, a framework for troubleshooting is established which enables isolating multiple faults. However, his method does not narrow down suspects by combining indeterminate inferences based on multiple test results.

5 DIAGNOSIS METHOD BASED ON TRUTH MAINTENANCE

5.1 Basic Constructs for Truth Maintenance

This section describes the basic constructs for truth maintenance and how they are utilized in diagnostic reasoning. A control method based on these constructs, specifying which assumptions to use in inferences and how to revise them according to discovery of contradictions, is described in the next section.

For easier context switching, the expert system adopts the formalism of ATMS(de Kleer 1986a; de Kleer 1986b), not that of TMS(Doyle 1979) or dependency-directed backtracking(Stallman and Sussman 1977), as a basic truth maintenance mechanism. Additional constructs, such as *justification*, *assumption*, and *nogood*, are introduced into the indeterminate diagnostic inferences.

According to newly derived inferences, *justifications* are created to record and maintain the relations between assumptions, facts and conclusions. The following is an example of justification.

```
premise_fact1, premise_fact2, assumption1
=> conclusion
```

In a justification, assumptions which make the inferences indeterminate, e.g. *assumption1*, need to be specified clearly. In order to realize the test omission, the anticipated result is also represented as an assumption, whose truth or falsity can be examined.

For discovering errors in inferences, contradictory conditions are also described. For instance, the following contradictions might occur in the reasoning:

(a) A conclusion *"The suspected units for the SymptomX are {X1,X2,...}."* contradicts the fact *"After replacing the suspected units {X1,X2...} with spares, there still remains the SymptomX."*

(b) An observation of intermittent occurrences of the symptom contradicts the assumption *"The symptom is not intermittent."*

(c) An assumption *"The fault is not related to the AreaX."* (which has been derived from the fact *"The fault is not observed in the AreaX."*) contradicts an observation *"The fault occurred in the AreaX"*.

Contradictions are represented by *nogood* in justifications. The following is an example for (b):

```
(observation of intermittence)
(assumption of non-intermittence)
=> nogood
```

Based on this formalism, justifications are created in the inferences for narrowing suspects, and then a chain of justifications is constructed. Fig-5 shows an example of a justification chain. The chain shows the relations between test results, assumptions and conclusions on suspects. The figure also includes two nogoods, one produced by {asmA, asmB, asmC} and one produced by {asmC}. By viewing this chain, underlying

assumptions which lead to any conclusions or nogoods can be listed. Of course, these relations are not represented exactly as in the figure but are represented in a data structure that enables fast context switching of ATMS.

5.2 Environment Control Method

With the basic formalism and constructs given in Section 5.1, a control method becomes a problem. When a contradiction is discovered, the set of believed assumptions must be revised to become consistent. Suppose there are no other contradictions and retracting any one of the contradictory assumptions removes the contradiction. Then, which assumption to retract becomes a problem. This section describes the control method.

In the following, the terminology of *environment*, *context*, and *nogood* in ATMS is utilized. In short, an *environment* is a set of assumptions. A *context* is a set of assumptions and nodes derivable from the assumptions. Thus, a context is characterized by an environment. *Nogood* is a set of contradictory assumptions.

With respect to the indeterminate inferences in the diagnosis, default assumptions are used to enable the *usual-case* inferences. Then, a control method which makes use of as many default assumptions as possible is desired. Therefore, when a contradiction is discovered, and consequently some assumptions are to be retracted, the number of default assumptions retracted should be kept as small as possible.

Suppose only a single environment is always chosen for reasoning, there arise problems in satisfying the previous desire. There seems to be no reasonable way to choose only one environment when assumptions have the same plausibility. In addition, environment control is not simple. A default assumption which has been already retracted to dissolve a contradiction may need to be made *in* again. Fig-6 shows a simple example for this. In the example, assumption *A*, which has been retracted to dissolve the nogood among assumptions *A*, *B* and *C*, is made *in* again in order to keep as many assumptions as possible *in*.

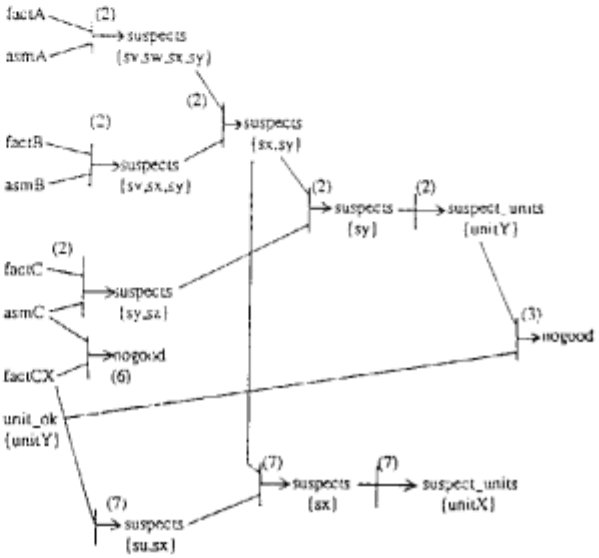
Considering these problems, the system does not determine only one environment, but realizes a control method which deals with multiple environments for reasoning. Reasoning in multiple environments is based on the ease of context switching of ATMS. For these reasons, the following is introduced:

Maximally Consistent Environment Set

The set consists of possible environments which satisfy the following:

- The environment is consistent. That is, the environment is not a subset of any nogood.
- The environment is not a subset of any other environments in the set.

If there are no contradictions, the set consists of an environment which includes all default assumptions. The set is updated according to the discovery of contradiction. Reasoning is enabled in any environments (sets of assumptions) in the set. By reasoning



factA : The symptom depends on the configuration.
asmA : The symptom is non-intermittent.
factB : The control order is distributed normally.
asmB : The clock works normally.
factC : A signal is not detected by a signal checking test.
asmC : The signal checker device works normally.
factCX: The signal checker device does not work.

Fig-5: Justification chain

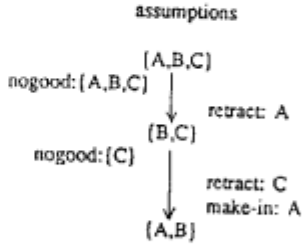


Fig-6: A complex case of environment control

in these environments, utilization of as many consistent default assumptions as possible is achieved.

When updating of environment sets due to contradictions, there arises the possibility of executing the omitted tests. Executing an omitted test for checking an anticipated result is effective in reducing the size of the maximally consistent environment set. This is introduced as:

Execution Control of Omitted Test

An omitted test, which is effective in reducing the size of the maximally consistent environment set, is defined as follows:

- Execution of the test has been omitted.
- The anticipated result of the test, which is an assumption, is included in some environments in the maximally consistent environment set and, at the same time, is not included in some other environments in the set.

The second condition implies that by the real truth or falsity of the anticipated result, several environments in the set can be eliminated from it. A test which satisfies these conditions should be executed when a contradiction is discovered.

5.3 Example

This section presents a simplified example of environment control. The example refers to the reasoning chain in Fig-5. Changes of the maximally consistent environment set, nogoods, and suspected units are described in Fig-7.

(1)Initial state:

All the default assumptions are *in*. Among them, *asmC* is an anticipated result of an omitted test and its truth or falsity can be checked.

(2)Narrowing suspects:

Several facts (test results) are obtained by test executions, and the suspects are narrowed down by indeterminate inferences. Accordingly, several justifications are created. Then, the conclusion that a defective unit is *unitY* is obtained.

(3)Unit replacement & Contradiction:

The suspected unit *unitY* is replaced. If the fault symptom does not disappear after this, this causes a contradiction.

(4)Contradiction dissolution:

According to the set of nogood assumptions, namely {*asmA, asmB, asmC*}, the maximally consistent environment set is revised.

(5)Execution of omitted test:

Because *asmC* is an assumption whose truth or falsity can be checked by an actual test, the omitted test is executed.

(6)Error of assumption:

Suppose the test result gives a fact *factCX* which is contrary to *asmC*. This causes a contradiction, and the maximally consistent environment set is revised. Environments including *asmC* are removed from the set, and only an environment {*asmA, asmB*} remains in the set.

(7)Another suspect:

Using *factCX* new inferences are made, and another suspect is obtained *(unitX)*.

5.4 Remarks on Truth Maintenance Incorporation

In the course of system development, the truth maintenance facilities described above have been incorporated into the former system. With this incorporation, the knowledge base for only usual cases has been modified into a more accurate version which clearly distinguishes the uncertainty in the indeterminate inferences.

By modifying the knowledge base, the diagnostic capability has been enhanced. The former system without truth maintenance could not continue diagnosis if the trouble did not disappear after replacing the suspected units. On the contrary, the modified system based on truth maintenance can infer other suspects by retracting assumptions.

Knowledge base modification for the incorporation was naturally achieved. This was achieved mainly by inserting the description of assumptions to represent

Phase	Maximal consistent environment set	Nogoods	Suspected units
1)Initial state	{{asmA,asmB,asmC}}		
2)Narrowing suspects			{unitY}
3)Unit replacement & Contradiction		{asmA,asmB,asmC}	
4)Contradiction dissolution	{{asmA,asmB},{asmB,asmC}, {asmA,asmC}}		
5)Execution of omitted test			
6)Error of assumption	{{asmA,asmB}}	{asmC} & {asmA,asmB,asmC}	
7)Another suspect			{unitX}

Fig-7: An example of environment control

indeterminate factors clearly and was relatively easy.

Based on the truth maintenance, some trial reasoning facilities are implemented as a flexible man-machine interaction facility. Based on a user-specified test result, the system hypothetically narrows down suspects and displays them. This helps the user understand the effectiveness of the test. These trial facilities are easily implemented in the truth maintenance framework.

6 CONCLUSION

This paper has presented the diagnostic method and the architecture of a troubleshooting expert system for electronic switching systems. Using a diagnostic method modeled on human expert strategies and reasoning, the expert system diagnoses faults which cannot be diagnosed by built-in functions. Using knowledge representation methods such as an abstract-signal-flow model and network-based knowledge representation, a compact and modifiable knowledge base is constructed.

The incorporation of truth maintenance enables effective diagnostic inferences when there is uncertainty. The environment control method which makes use of default assumptions realizes this kind of reasoning. The method also enables flexible diagnosis, like inferences based on anticipated test results and later execution of actual tests when doubtful. Incorporation of truth maintenance greatly enhances the diagnostic capability and flexibility of the system.

ACKNOWLEDGMENTS

The authors would like to thank Mr. Y. Fujii, of the Institute for New Generation Computer Technology for various kinds of support. The authors express deep appreciation to Mr. Y. Souda, Dr. S. Goto, and Mr. H. Mori for continuous encouragement and support. The authors thank Mr. K. Toudou, Mr. H. Igarashi and their colleagues for their patient cooperation during the knowledge acquisition study. Finally, the authors thank Mr. N. Wakasugi, Mr. M. Oishi, Mr. T. Arai, Mr. H. Kijima, Mr. S. Takahashi, Mr. T. Hisada, Mr. K. Satozaki, and Mr. M. Ikeda, for the implementation of the expert system. This research was carried out as a part of the Fifth Generation Computer project of Japan.

REFERENCES

(Chikayama et al. 1984)
T. Chikayama, S. Takagi, and K. Takei, ESP - An Object Oriented Logic Programming Language, *ICOT Technical Report TM-0075*, Institute for New Generation Computer Technology, 1984.

(de Kleer 1986a)
J. de Kleer, An Assumption-based TMS, *Artificial Intelligence 28*, 1986, pp.127-162.

(de Kleer 1986b)
J. de Kleer, Problem Solving with the ATMS, *Artificial Intelligence 28*, 1986, pp.197-224.

(de Kleer and Williams 1987)
J. de Kleer and B. C. Williams, Diagnosing Multiple Faults, *Artificial Intelligence 32*, 1987, pp.97-130.

(Doyle 1979)
J. Doyle, A Truth Maintenance System, *Artificial Intelligence 12*, 1979, pp.231-272.

(Shortliffe 1976)
E. J. Shortliffe, *Computer Based Medical Consultations: MYCIN*, Elsevier, New York, 1976.

(Stallman and Sussman 1977)
R. M. Stallman and G. J. Sussman, Forward Reasoning and Dependency-directed Backtracking in a System for Computer-aided Circuit Analysis, *Artificial Intelligence 9*, 1977, pp.135-196.

(Taki et al. 1984)
K. Taki, M. Yokota, A. Yamamoto, H. Nishikawa, S. Uchida, H. Nakashima and A. Mitsuishi, Hardware Design and Implementation of the Personal Sequential Inference Machine(PSI), *Proc. of the International Conference on Fifth Generation Computer Systems*, 1984, pp.398-409.

(Young 1987)
M. Young, A Framework for Describing Troubleshooting Behavior Using Default Reasoning and Functional Abstraction, *Proc. of the Third Conference on Artificial Intelligence Applications*, Hyatt Orlando, February 1987, pp.260-265.