Order Number 8802345

Development of expert systems as on-line operational aids

Tomsovic, Kevin Louis, Ph.D.

University of Washington, 1987

U·M·I

300 N. Zeeb Rd.
Ann Arbor, MI 48106

DEVELOPMENT OF EXPERT SYSTEMS AS ON-LINE OPERATIONAL AIDS

by

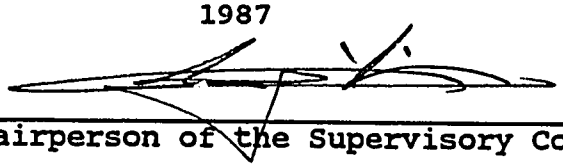
Kevin Tomsovic

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Washington

1987

Approved by 

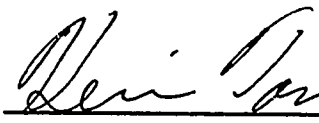
(Chairperson of the Supervisory Committee)

Program Authorized
to Offer Degree Electrical Engineering

Date June 10, 1987

Doctoral Dissertation

In presenting this dissertation in partial fulfillment of the requirements for the Doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to University Microfilms, 300 North Zeeb Road, Ann Arbor, Michigan 48106, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature 

Date June 10, 1987

University of Washington

Abstract

DEVELOPMENT OF EXPERT SYSTEMS AS ON-LINE OPERATIONAL AIDS

by Kevin Tomsovic

Chairperson of the Supervisory Committee:
Professor Chen-Ching Liu
Department of Electrical
Engineering

In recent years, the interest in Expert Systems (ES) has grown rapidly. The application of expert systems to power system operations and on-line environments is a new area of research. In this thesis, the issues involved with on-line knowledge based systems for operations is investigated. To begin, the development and validation process involved in building two ES operational aids are discussed. One incorporated logical and heuristic rules for Customer Restoration And Fault Testing (CRAFT) of power lines with automatic switches. The other, referred to as VCES (Voltage Control Expert System), integrated numerical and heuristic techniques for reactive power/voltage control in a power system. Both expert systems are implemented as rule-based forward-chaining systems.

Can expert systems such as CRAFT and VCES be applied in an on-line operation environment? The answer depends on the quality of the knowledge base and the computational efficiency. Besides validating the knowledge base, computation time is analyzed. Based on discrimination nets,

used in many production system languages for pattern matching efficiency, a method is proposed to obtain the worst case computation time of a rule-based system. The method depends on bounding the individual computational components of the inference engine (pattern matching, conflict resolution, and actions) and partitioning rules into groups performing different tasks in the decision-making. The worst case time is obtained by combining the costs of each of these tasks. This research is a step toward insights into the application of knowledge based systems to power system operations and the efficient implementation of rule-based systems.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

This research was sponsored principally by Electric Power Research Institue, Palo Alto, CA, and Puget Sound Power and Light (PSPL), Bellevue, WA. However, much of my graduate work was sponsored by the Eldec Corporation, Seattle, WA. The author wishes to thank these organizations. Additionally, the author expresses sincere appreciation to the members of the examining committee, Professors C.C. Liu, M. Damborg, A. Holden, E. Noges and A. Borning. Particular gratitude is due to Prof. C.C. Liu for his untiring efforts which have improved the quality of this thesis immeasurably. Several people provided assistance in the programming efforts and deserve recognition. George Shraw worked on the simulator portion of CRAFT and input PSPL line data into the database. Paul Karr also worked on the simulator. Shijie Zhang translated part of the OPS83 code from OPS5 code for VCES. Thanks also to several dispatchers at PSPL for their valuable input but particularly Paul Ackerman, Jim Dodge and Steve Pope.

Finally, but certainly not the least important, I wish to identify and thank several friends (Aggoune, Ashutosh, Brian, Bruce, Christine, Doreen, Eric, Greg, John, Hemant, Kathleen, Kathleen, Mani, Mingliang, Mo, Mohammed, Nan, Naushod, Rich, and Shira among others) who have added little to this thesis but greatly to my well-being.

We can forgive a man for making a useful thing as long as he does not admire it. The only excuse for making a useless thing is that one admires it intensely.
All art is quite useless.

Oscar Wilde

We had to wait four billion years but we finally got Jerry Lewis, MTV, and a large selection of creme-filled snackcakes.

Zippy the Pinhead
(Bill Griffith)

1. INTRODUCTION

An Expert System (ES) may be defined as [1]:

. . . *an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant expertise for their solution.*

Several other definitions exist, however, the essential idea is to separate knowledge of the application domain from the inference procedure and data in order to allow incremental improvements within the knowledge base effectively simplifying development. Typically, knowledge is encoded in rules or some other highly structured representation (e.g. frames). Among the best known of the early ES is MYCIN [2] developed for performing medical diagnosis and R1 [3] developed for configuring VAX computers. The potential areas of application are extensive [4].

The application of ES in on-line environments is a new area of research. (On-line will mean that the ES is performing in some sort of operational environment, for example, computer system operator [5], air traffic controllers [6], etc., but not necessarily a closed loop system). Additionally, ES applied to power system problems is a relatively new area of research. Some example areas are

alarm processing [7], system diagnosis [8,9], restoration and control [10,11,12,13,14]. This study deals with the application of ES as on-line power system operational aids. In the modern power system control center, on-line operations are characterized by a time constraint imposed on the response and an environment where potentially confusing amounts of data and alarms may be presented to the dispatcher, particularly during emergencies. Thus, it is seen that improved computing assistance for dispatchers would be valuable. The contributions of this study are: initiation of new ES applications in power system operations, development of two expert systems for on-line operations, and proposal of methodologies for performance evaluation.

An expert system was developed for assisting dispatchers in a control center when line sections become faulted and automatic switching is insufficient to isolate the fault and maintain service to all customers on the line in question [15]. The specific problem, identifying and isolating the faulted section in a multiple-tapped line, was suggested by dispatchers at Puget Sound Power and Light (PSPL) Company, Bellevue, Washington. At present, in case of a fault, dispatchers refer to diagrams of the faulted lines and perform a reasoning process based on the design principles of protective devices (breakers, automatic switches) and practical experience to determine the fault location. If it

is not possible to locate the exact faulted section by reasoning alone, dispatchers may attempt test actions (opening switches and/or closing breakers). The test actions are selected so as to restore the maximal number of customer substations if the actions are successful. However, if a test should fail, the SCADA system will acquire more information about the fault and therefore dispatchers can continue the diagnosis. Generalizations of ideas gleaned from discussions with dispatchers resulted in the formulation of the empirical knowledge encoded in the system. A rule based representation was chosen since this knowledge could be expressed naturally in terms of rules. A complete description of the developed system, named CRAFT (customer restoration and fault testing), is presented in Chapter 2.

Another proposed expert system application is concerned with the reactive power/ voltage control [12]. In the process of development, several extensions to this work were made [17]. For ease of reference, the proposed expert system will be referred to as VCES, Voltage Control Expert System. The rule base of VCES incorporates empirical rules for selecting controls, e.g., shunt capacitors, tap changers, generator excitation, in order to resolve violations of load bus voltage limits. For severe voltage problems, it was proposed to utilize an optimization method such as a reactive dispatch algorithm [16]. These methods are applied

to a test system to solve a number of scenarios of varying severity. The numerical and heuristic approaches are compared in terms of the quality of solution and the computational speed. The expert system processing is broken down to a number of basic tasks, for which the computational costs are measured. Since the measurements are designed to be problem independent, they are used to estimate the processing costs of these tasks for the selected voltage problem scenarios. Chapter 3 presents the detailed development and performance analysis of VCES.

Up to the present, there have been no reports on the implementation and operational experience of expert systems in an Energy Management System environment [18]. Can expert systems be applied as an on-line operational tool? Two critical issues need to be addressed: (1) Is the ES capable of solving the specific problem that it is designed for? (2) Is the rule-based program efficient enough to solve the problem within the available time? The first question depends on a high quality knowledge, while the second one has to do with the implementation. Notable implementation issues include rule structure, data structure, software language, and hardware support.

Due to the inherent uncertainty in heurisitcs and the complexity of rule-base programs, the validation of an expert system is nontrivial. System operators or other human experts can evaluate results of various simulation cases.

The purpose is to validate the knowledge, reasoning process, and computational efficiency. In chapters 2 and 3, extensive simulations are used to verify the knowledge base and measure computational requirements. It is important that the simulations fully exercise the strengths and weaknesses of the developed expert systems. Chapter 4 will address the issue of computational efficiency more thoroughly. An expert system may be quite time-consuming since it must search the entire rule-base before every rule firing. From the computational point of view, it is desirable to identify the bottleneck of expert system processing. For example, a poor rule formation can cause extra rule-chaining and therefore slow down the decision process. An unnecessarily large rule base or large numbers of irrelevant data also increase the search time. The proposed approach is to determine an upper bound on the time required to solve any single scenario of the problem domain. (In the examples presented, this corresponds to one faulted line section or one problem voltage scenario.) In chapter 4 the following items are discussed: an upper bound on computations, an algorithm for the computation of this bound and an approach for identifying bottlenecks of the rule-base. Numerical examples are presented.

2. EXPERT SYSTEM APPLICATION TO FAULT DIAGNOSIS AND CUSTOMER RESTORATION

Faults may occur in transmission lines due to stormy weather conditions, traffic accidents, etc. Typically, each line is equipped with power circuit breakers on both sides which are designed to trip within cycles after a fault occurs in order to protect equipment from damage. Transmission lines in the 115KV range often have substation taps between these power circuit breakers. A tripping of the breakers will lead to interruption of customers supplied through taps along the faulted line. To minimize the power interruption during the line faults, most circuit breakers are designed with timed reclosures for automatic restoration in the case of temporary faults (e.g. blowing trees, lightning, etc.). Additionally, an effective scheme is to install automatic switches between some sections of the line. These switches are designed to close or open depending on the status (not or dead) of the component (line or bus). Some types of the automatic switching are: Open Dead Line (ODL), Close Dead Line (CDL), Close Hot Line (CHL), Open Dead Bus (ODB), Close Dead Bus (CDB), Automatic Transfer Dead Bus (ATDB) and Automatic Transfer Dead Line (ATDL). The switches are provided with a delay time setting for the operation. For example, ODL + 4 seconds means that the

switch is set to open four seconds after a dead (de-energized) line is detected. System designers can coordinate the time settings among switches in order to achieve minimal power interruptions under specified fault conditions.

Upon the occurrence of a fault, the operation of supervised automatic switches will result in alarms presented to system dispatchers. Unsupervised automatic switches require communication from personnel in the field before a dispatcher has an indication of the device operation. In practice, dispatchers need to examine the system configuration following the fault and possibly they will need to perform a set of test actions to determine the faulted section. The decision-making process involves reasoning based on system status, functions of automatic switches and empirical judgments related to specific line conditions. Since the dispatchers also have to retrieve system data from handbooks and/or computer storage, the decision-making process can be time-consuming, depending on the complexity of the line and the fault event.

This problem solving approach followed by the dispatchers, i.e. heuristics, operational guidelines, etc., suggests an expert system solution. The organization of this section is as follows: the computing environment for the expert system is discussed in Sec. 2.1. Sec. 2.2 presents the knowledge gleaned from system dispatchers for analyzing problems arising from line faults. Sec. 2.3 shows the

details involved in implementing the gathered knowledge. A brief comparison between the production system languages OPS83 and OPS5 is made. Practical examples are presented to demonstrate the capability of the developed expert system.

2.1 Expert System in Operating Environment

The structure of an expert system, i.e. knowledge contained in rules, is such that it provides capabilities and features which are difficult for conventional programming techniques to achieve. These capabilities and features have been extensively discussed in the literature [e.g. 4,20]. A number of expert systems for power systems have been developed [7-15]. However, the issue of embedding these expert systems in a modern control center has not been thoroughly investigated. Existing control centers typically have a number of mainframes, a customized data base and a number of application programs written in traditional languages, such as FORTRAN. An expert system can be most easily written in Artificial Intelligence languages such as LISP, Prolog, OPS5 and OPS83. Thus, software compatibility is an issue. When compilers or interpreters of the required AI languages are available for the control center hardware, then computational efficiency and the design of appropriate interfaces to existing software must be considered. It may also be possible to make specialized hardware additions, e.g., one approach to implementation for PSPL is to use a

microcomputer and an interface between the microcomputer and existing mainframes.

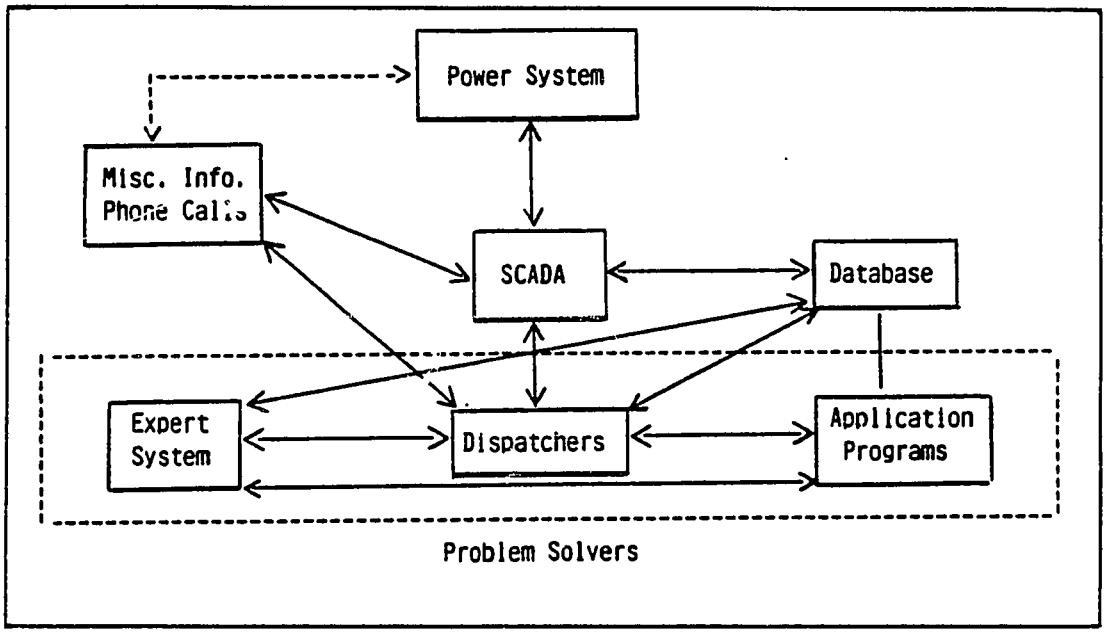


Fig. 2-1 Expert System in Computing Environment

Fig. 2-1 shows the structure of the proposed expert system within the modern control center computing environment. The system has been written in a production system language, OPS83 [21]. Specialized production system languages, such as OPS83, provide many features which facilitate development of an expert system. However, the high level of these languages may present difficulties when communicating with other software. As mentioned previously, other computer support, such as the SCADA system, is typically written in more conventional languages, e.g. on-line code in assembly language and applications programming in FORTRAN. Communication between these various processes

must be established. For the application of this study, it is proposed that the expert system views the power system by monitoring the database. As alarms which indicate line faults are posted, the expert system gathers the most recent data and begins to work on the problem with dispatchers. While the proposed expert system is not actually implemented in the control room, this issue of interfacing an expert system to a database is addressed. The expert system has been developed on a VAX 11/780 processor using the VMS operating system. Extensive discussions on database management systems (DBMS), and the literature contained within, can be found in [22,23]. The interface to the database will be discussed in light of the expert system development.

2.2 Knowledge Acquisition Results

In this section, we present the knowledge incorporated into the expert system. It is important to develop the expert system in as general a framework as possible for several reasons. To begin, portability to other power systems can be achieved. Extensions of the current solution approaches or expansions into new problem areas will be straightforward. Furthermore, general rules simplify development in that fewer rules are needed to govern a class

of problems.

Discussions with Puget Power dispatchers have identified the reasoning and several heuristics which are typically utilized in solving the fault isolation problem. A few assumptions are needed, and are listed here:

a) All power circuit breakers operate according to design and thus isolate the line fault to within a single line, i.e. the possibility of miscoordination is not considered.

b) All other automatic switching devices function normally as well.

c) All faults are single line faults, i.e. no bus faults and not more than one faulted section on any particular line.

d) All switches and breakers have reached their final configuration, i.e. any automatic switching operations which are going to occur have already taken place.

e) Through SCADA or communication with field personnel any device openings or closings are known to the dispatcher and updated in the system database.

These assumptions define the most common scenarios that PSPL dispatchers face. As such it was felt that assistance during these operating conditions would be the most beneficial. In the following, we identify a number of the

empirical rules our system is based on:

a) The line fault must lie between open breakers or switches.

b) For any unsuccessful breaker reclosures, the fault must lie between the breaker and the nearest open device at the instant the reclosure is attempted. For the indicated fault in Fig. 2-2, breaker 6 will attempt to reclose 16 seconds after the fault occurs. Before this attempted reclosure occurs, ODB switch 4 will have opened, the breaker reclosure will be unsuccessful, which implies the fault must lie between devices 4 and 6.

c) If any customer is without power and the fault location is not known precisely, then in order to obtain further information about the fault location, testing of the line, i.e., opening or closing breakers and/or switches to attempt to restore the customer, may be considered.

d) Test closing a breaker or a switch should not be done if a failure will interrupt service to customers currently with power.

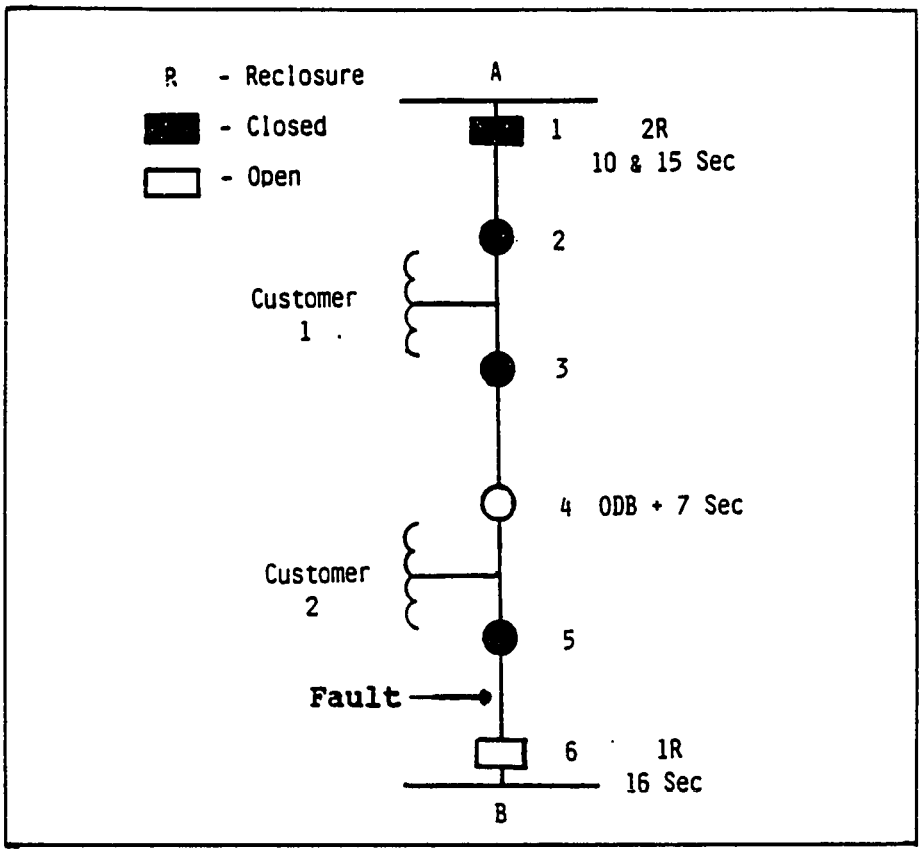


Fig. 2-2 Example System for Item (b)

e) As many customers as possible should be brought on line by any successful test. In Fig. 2-3, one has several alternatives for testing the line. For instance, suppose one decides to restore customer 1 and customer 3 by opening 3 and 6 and closing 1 and 8. Then, the situation exists where an unsuccessful attempt to energize customer 2 through either device 1 or 8 will interrupt service to a customer. Such an interruption violates the previous rule, (d). According to rule (e), one should attempt to

bring back all 3 customers in the first test, i.e. open 7 and close 1 or open 2 and close 8.

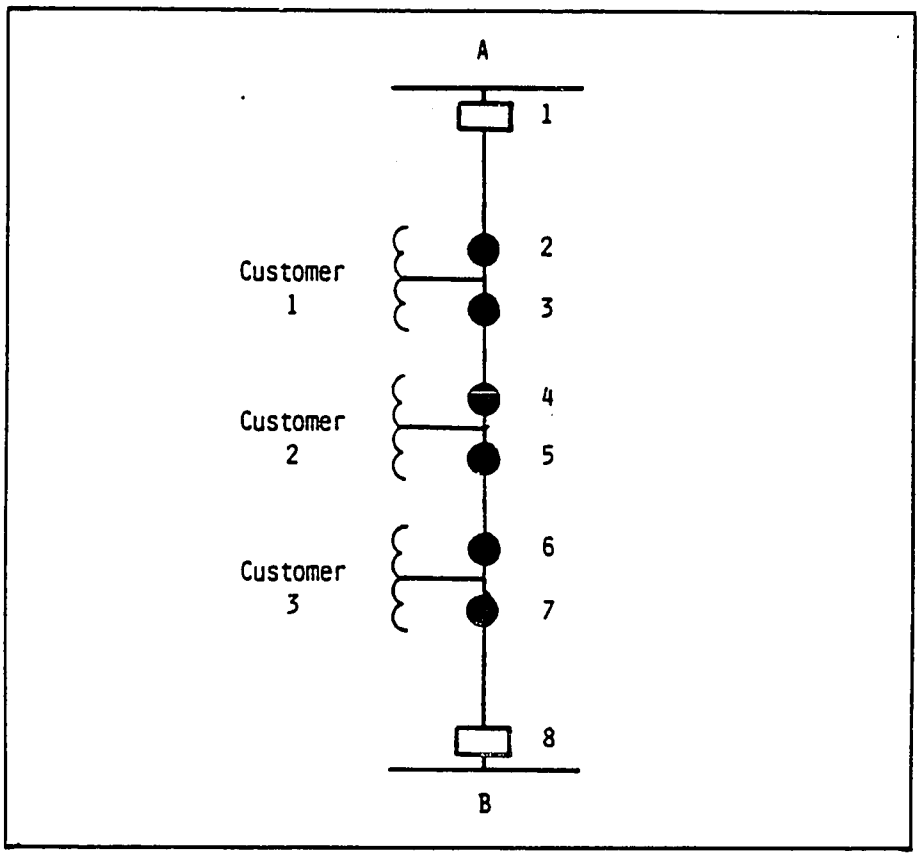


Fig. 2-3 Example System for Item (e)

f) Stormy weather may result in a significant number of line outages. In order to maintain strong ties to the power system, dispatchers may be more aggressive in testing lines to avoid vulnerable system configurations, including risking interruption of power to customers currently with service (rule (e)). It is assumed that weather information is available.

g) When energizing a line which may be faulted, it is preferable to close a breaker into the line rather than switching in load breaks or section switches which, in general, are more likely to be damaged by closure into a fault.

h) When attempting to restore a substation, supply power from the "stronger" source(end) of the line available. The strength of a source is a heuristic judgment based on the amount of power typically delivered through that end of the line. The stronger side often has more breaker reclosures, in an attempt by planning engineers to keep this source in service. Consider the line in Fig. 2-4, it has only circuit breakers for automatic switching. For any fault that occurs on this line, a dispatcher will have the choice of supplying power through either of the two breakers. In the absence of other information, switch 5 would be opened and breaker 1 closed (instead of opening 2 and closing 6) so that if possible customers are picked up through the stronger tie at A.

i) Substations usually have isolation switches (e.g. switches 2 and 3 in Figs. 2-2, 2-3, 2-4). The relatively short distance between these devices makes it highly unlikely for a fault to occur on the section of line connecting them.

j) 3-terminal lines can be analyzed by dividing the line into 2-terminal line sections, then selecting one of

these sections and proceeding as before. (A 3-terminal line is terminated by three power circuit breakers which encompass a branching point on the line, see Fig. 2-5).

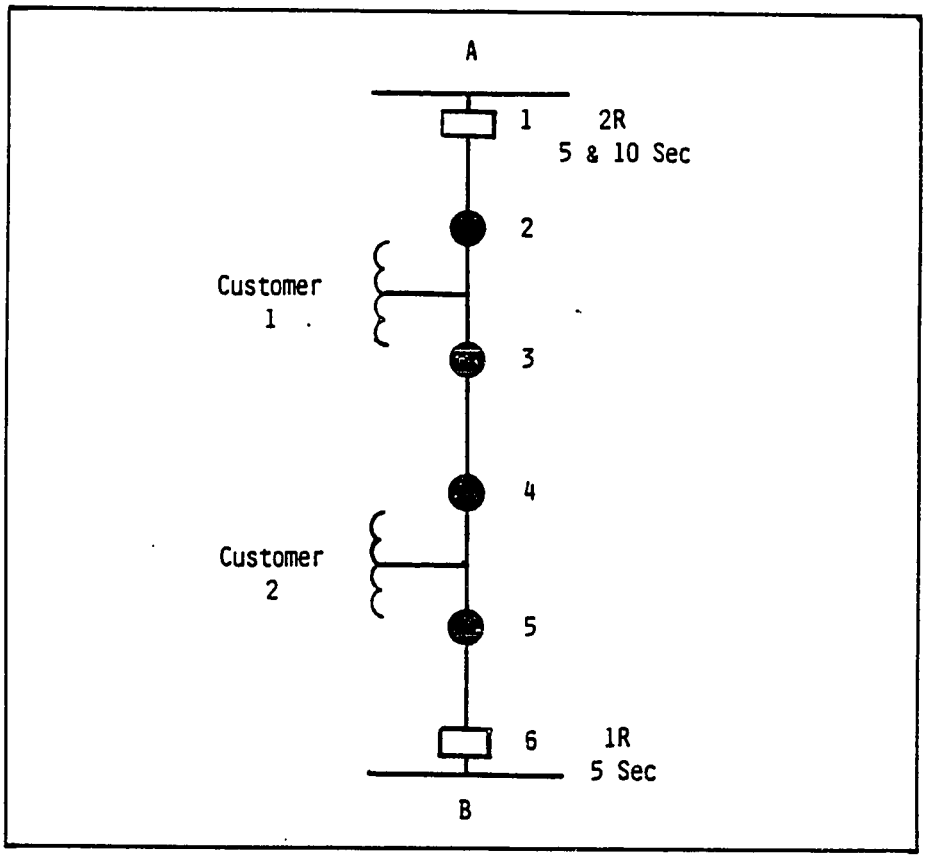


Fig. 2-4 Example System for Item (h)

k) Within a 3-terminal line, branches with substations are considered first for purposes of locating the fault. In Fig. 2-5, barring other information, the line sections between B and C are considered first.

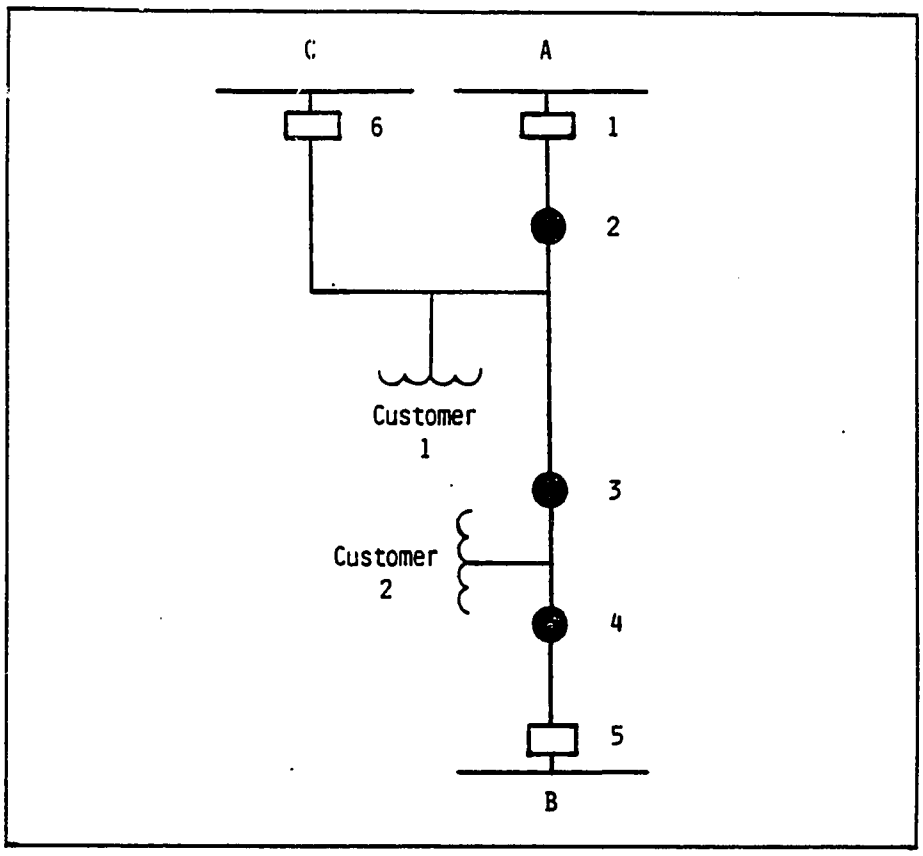


Fig. 2-5 Example System for Item (k)

l) Closing a switch into a fault puts at risk of personal injury any persons who might be in the vicinity of the line in question. Safety is of top priority.

m) Other specific line conditions affect the priority of line testing. For example, faults on lines that run through busy intersections are often the result of car-pole accidents. Thus, it is likely that there will be people in the immediate area and testing would not be appropriate.

n) ODL and CHL devices are designed with a 'lockout'

feature to avoid cycling between open and closed indefinitely. A lockout indication provides further information about switching operations that took place and thus about the possible fault location.

o) A variety of miscellaneous information is known to dispatchers, including behavioral information about specific power system components, SCADA operation, etc.

The proposed expert system also contains knowledge so as to enable dispatchers to work on several problems simultaneously. That is, the dispatcher can suspend work on a given problem in order to analyze other alarmed lines. The above empirical rules along with procedural information extracted from dispatchers form the knowledge base which has been written as approximately 200 production rules.

2.3 Expert System Development

In this section, we describe the design of the developed expert system. Sec. 2.3.1 presents the programming strategy used for implementing the knowledge presented in Sec. 2.2. Relative merits of the productions system, OPS83, language is discussed in Sec. 2.3.2. An approach for interfacing to a database is shown in Sec. 2.3.3. The computational efficiency of the system is considered in Sec. 2.3.4 and finally the expert system validation is discussed.

2.3.1 Rule Base Organization

Consistent with the goal of generality and ease of modification for expert systems, certain techniques are employed in implementing the aforementioned knowledge. For instance, the overall logical flow of the developed expert system is to form a hypothesis of a single line fault and then validate and analyze this conjecture further through reasoning and/or physical testing of the power system. In addition to this top-level framework, rules are grouped under tasks (or goals). Table 2-1 identifies these tasks for the proposed expert system while Fig. 2-6 gives an indication of how processing of a problem might proceed. Rules under each task are selected until the task is completed or until no more rules are applicable.

Under the first task 'process alarms', one can see in Table 2-1 four subtasks: classify problem, resolve alarms, approve proposed actions and perform bookkeeping. The 'classify problem' subtask determines that the alarms being received indicate a line problem. This information is passed to 'form hypothesis', Task 2, which will initiate the search for a solution. 'Approve proposed actions' allows the dispatcher to examine a set of actions the expert system recommends before actual implementation of a solution for the problem is to take place. Once a solution has been implemented, the system removes those alarms that can be accounted for by the determined fault. Unresolved alarms are

Table 2-1 Tasks in Rule Base

TASK 1: Process Alarms Subtasks: Classify Problem Resolve Alarms Approve Proposed Actions Perform Bookkeeping
TASK 2: Form Hypothesis
TASK 3: Validate Hypothesis Subtasks: Gather Data Formulate Data Locate Fault Set Up Testing Analyze Test Results
TASK 4: Check Timing
TASK 5: Test Line
TASK 6: Suggest Solution

shown to the dispatcher for further consideration. 'Bookkeeping' creates a log of information for utility records.

The formation of a hypothesis creates the third task 'validate hypothesis'. 'Validate hypothesis' contains the bulk of the empirical rules presented in Sec. 2.2 The first step is to gather relevant data, switch operations, substations on the line, etc., and formulate the data. If the line of interest is a 3-terminal one a 2-terminal line representation is constructed. When data has been gathered, control is passed to the subtask 'locate fault'. 'Locate fault' is the central focus for verifying the line fault hypothesis. When automatic switching is present 'locate fault' posts Task 4, 'check timing operations', which is used to further narrow down possible fault locations. As down possible fault locations. As seen from empirical rule (c), testing may be necessary if the fault is not located exactly. If testing is necessary, subtask 'setup test' determines which switches or breakers need to be opened or closed. Once all actions for testing have been determined and approved by the dispatcher 'test line', Task 5, performs the test under the constraints on safety procedures and obtains the results of the testing. The remaining procedure under 'validate hypothesis' is to analyze the test results. This analysis will also narrow down the possible fault locations. It may happen that more testing is needed

depending on the complexity of the event. If at any point in the process of looking for a single line fault an inconsistency is found, e.g. all possible locations are eliminated, a breaker fails to respond to a SCADA command, etc., the hypothesis is deemed invalid. If no inconsistency is found and the fault has been located the 'validate hypothesis' task is complete.

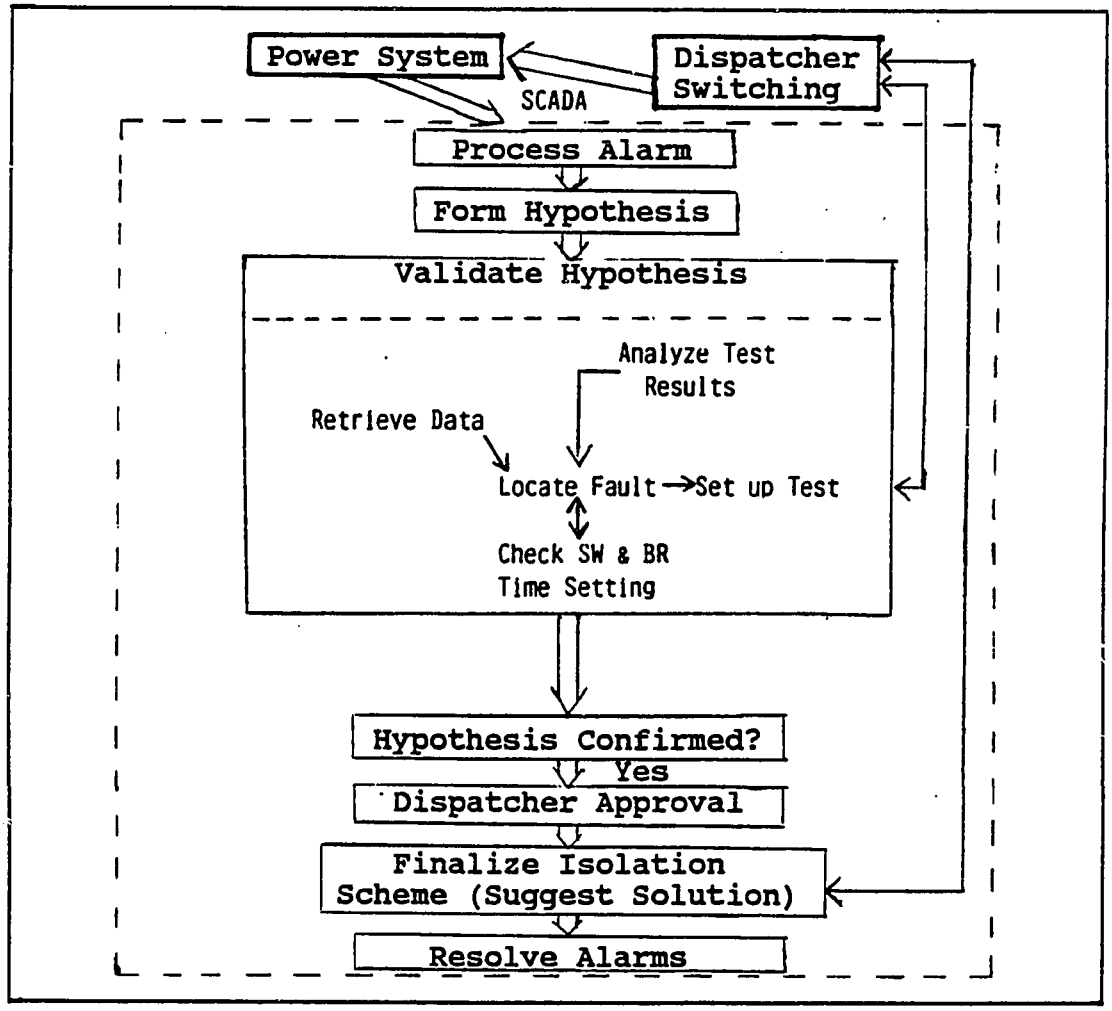


Fig. 2-6 Solution Process in Fault Isolation

Task 6, 'suggest solution', will determine any further

actions which must be performed to complete isolation of the fault. A complete set of actions is presented to the dispatcher which the expert system believes will solve the problem, dispatchers can utilize these results or rely on their own judgment. The task 'test line' requires implementation of switching actions by the dispatcher.

Notice, in the course of working on a task it is often necessary to create subtasks which define corresponding subproblems. By successively creating tasks and subtasks the problem can be solved in stages similar to the process made use of by dispatchers. In summary, two types of rules are embedded in the system. One, those rules directing logical flow, i.e. removing or establishing tasks. Two, those rules which represent behavioral information and form conclusions. Fig. 2-6 shows how the solution process flows in the proposed expert system. One of the main advantages of expert systems is the flexibility in the solution process. In this respect, Fig. 2-6 could be misleading in that it appears that the solution process follows step-by-step, similar to that in a FORTRAN program. Actually, in a production system the solution process is data driven. In other words, at each step the rule to be selected, and thus the action to be taken, depends on the current working memory elements. The whole rule base will be searched (in an efficient manner) to find the specific rules that match the working memory data. To emphasize this, a view of the logical flow of the system

is shown in Fig. 2-7, where a conflict resolution scheme is required to select the most applicable rule.

Portability is accomplished by incorporating general principles into the rule base, leaving utility specific information in the data base to the extent possible.

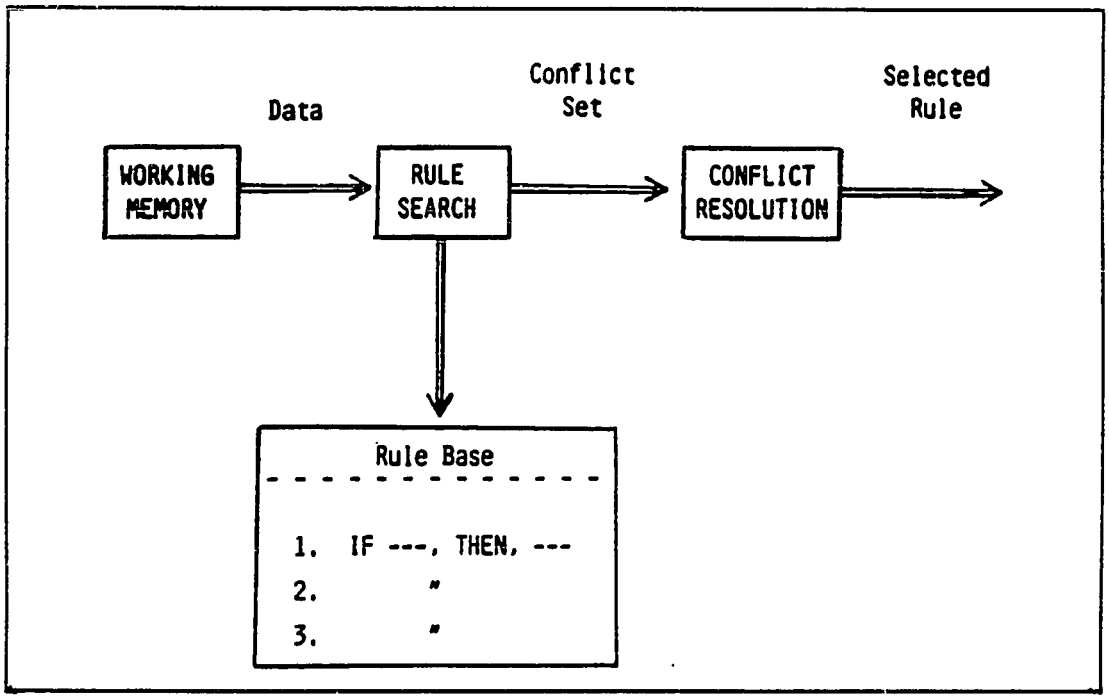


Fig. 2-7 Data-Driven Mechanism in Rule-Based Systems

2.3.2 Selection of a Production System Language

Specialized production system languages such as OPS5 and OPS83 are effective tools for developing expert systems [4]. They provide an inference engine, debugging routines and a fast pattern matching (rule searching) algorithm. In this study, OPS83 has been selected due to its flexibility and improvements over its predecessor OPS5. Major improvements

are briefly summarized here [21]:

1. OPS83 has simplified interfaces to subprograms written in other languages such as FORTRAN. This would be a major advantage in power system applications.

2. According to information from the distributor, OPS83 is 30 times faster than the LISP based version of OPS5 and 4 to 5 times faster than the BLISS based version of OPS5.

The syntax of OPS83 rules is similar to that in its predecessor OPS5. An example rule is given below in English. The OPS83 implementation is given in Fig. 2-8.

Rule 1

IF the task is to locate a single fault
AND the single fault hypothesis is being tested
AND the fault is on a radial section of line
AND a substation is within the possible faulted sections
AND no automatic switching lies within the possible faulted sections
THEN modify task to set up test switching

```
&task (task id=test_hypothesis; status=locate_fault;);
&hypo (hypothesis id=single_fault;    status=test;);
&fault (line_fault status=radial;);
&sub (substation status=out; from>=&fault.start;
                 to<=&fault.end;);
  -(device automatic=yes; location>&fault.start;
                 location<&fault.end; );
-->
modify &task (status=setup_test; );
```

Fig. 2-8 OPS83 Implementation of Rule 1

2.3.3 Database Interface

Power system control centers often have customized database systems. To address the database issue of the expert system implementation we have chosen to use a relational database due to advantages of data independence, community view of common data, and simple data access in application programs [22]. In particular, the relational database language used is UWRIM (University of Washington Relational Information Management).

An approach for communicating between the database and the expert system has been developed. The essential problem in passing data to the expert system arises from different data structures available in OPS83 and FORTRAN (UWRIM is FORTRAN based). More specifically, symbolic (alphanumeric) data is incompatible. In FORTRAN, character strings of fixed lengths provide the basis for symbolic data. OPS83 relies on variable length character strings called symbols. As a simple example, in FORTRAN the switch status information 'closed' or 'open' could not be assigned to the same variable without adding blank characters as filler, i.e. 'closed' and 'open '. OPS83 requires no such character padding. Thus for names and other descriptive information, some data translation must be performed. An efficient translation could be accomplished by performing low level operations on data. Alternatively, and the approach taken in this project, software may translate symbolic information into a standard format readable by both programs. The

simplest method is to use a temporary text file. Since all programming languages provide standard input/output procedures for accessing data in sequential text files, this approach affords greater clarity and portability than a low-level approach. With the issue of compatibility resolved, data can be retrieved from the database by calling subroutines which read from and write to a text file as a means of communication.

2.3.4 Efficiency Issues - Initial Considerations

The operator assistant we have proposed is intended for an on-line system and consequently time efficiency is of importance. Keys are used in the database to improve data access time. OPS83 itself has an efficient algorithm for pattern matching to enhance time performance. CPU time consumed in reaching a solution was measured to indicate the efficiency of the developed system. When there are a number of lines which are faulted, it is important to consider how dispatchers approached the problems. For example, it is more time consuming to alternate the line fault which is under analysis. In our system, if one problem is worked to completion before considering other lines, the CPU time required for each problem was around 2-3 seconds with approximately 1-2 seconds of this time spent retrieving data from the database. More details on the computational efficiency will be presented in Chapter 4. It is important

to note that for large systems the database access time will increase but since the consideration of line section fault problems is restricted to one line at a time, the time spent analyzing the fault will remain small. In other words, for large scale systems the time bottleneck is expected to be the database access.

2.3.5 Performance Validation

It is often more difficult to evaluate the performance of knowledge based systems than conventional algorithmic software. For our system, a rule-based simulator of about 75 rules has been built to facilitate the testing. Dispatchers are relied on to evaluate the response of the expert system to various simulated fault scenarios.

2.4 Simulation Results

Several case studies are presented to illustrate the capabilities of the developed expert system. In the first example, a detailed analysis of the solution methodology and the required empirical knowledge are identified for clarification. For other examples, only the problem and the proposed solution are provided here. In all cases, any suggested actions require the dispatcher to implement the action through SCADA or field personnel.

Table 2-2 Notation for Automatic Switches

ATDB	- Automatic Transfer Dead Bus
ATDL	- Automatic Transfer Dead Line
CDB	- Close Dead Bus
CDL	- Close Dead Line
CHL	- Close Hot Line
HLR	- Hot Line Reclose
NO	- Normally Open
ODB	- Open Dead Bus
ODL	- Open Dead Line
R	- Automatic Reclosing
S	- Supervisory
⊂—	- Sensor

Case 1 (Fig. 2-9)

Faulted Section: The line section between switches 874 and 485.

Final configuration of protection devices:

Circuit breaker 361 open

Load break 483 open

Load break 794 open

All other devices are closed

Analysis: This example illustrates a variety of capabilities in the present system. The 'classify problem' task identifies the presence of a line fault due to alarms on the opening of protection devices. The single line fault

hypothesis is then formed. The first step in analyzing the problem is to retrieve data about the line in question. In this case, a 2-terminal line needs to be constructed. Neither the branch to Starwood nor to White River contains substation taps, thus either branch can be chosen and added on to the section leading from O'Brien to load break 422. The branch to White River is chosen arbitrarily. Now, control passes to the 'locate fault' subtask. The fault is first identified as lying somewhere between breaker 361 and load break 794 (empirical rule 'a'). Note that load breaks 794 and 483 open on a dead bus. Both devices would be open before the time of the second attempt to reclose 361. The 'check timing operations' task applies empirical rule 'b' and deduces that the fault must lie between breaker 361 and load break 483. 'Locate fault' eliminates as a possible fault the line section between switch 485 and load break 483 since it is highly unlikely (empirical rule 'i') that this section will be faulted. Testing is deemed necessary in order to determine whether to supply power to the industrial customer Weyerhauser from White River/Starwood or from O'Brien (empirical rule 'c'). The system suggests supplying power from O'Brien as a test.

If the attempt had been made to supply power from White River and the test failed, power would be interrupted for customers Belmore and Kitts Corner thus violating empirical rule 'd'. In this scenario, the test from O'Brien would be

successful. Subtask 'analyze test results' then eliminates the line sections between breaker 361 and switch 874 as a possible faulted section since these sections have been successfully re-energized. At this point, the fault location has been narrowed down to the section of line separating switches 874 and 485. With this much known about the fault location, a solution can be suggested which will restore all customers on the line. 'Suggest solution' indicates that load breaks 483 and 794 can be closed to pick up substation Marine View. The dispatcher has the option to accept or discard this result. If the closures of 483 and 794 are implemented, 'test line' serves to detect unexpected system response which may result from inaccurate fault analysis. Lastly, any required reports of the analysis are created from the fault location information. Updates to the database are made to show this line section is under maintenance. In summary, the proposed actions are:

Testing:

Open section switch 874

Close breaker 361

Test successful

Final solution: The following actions should be taken sequentially:

Open section switch 485

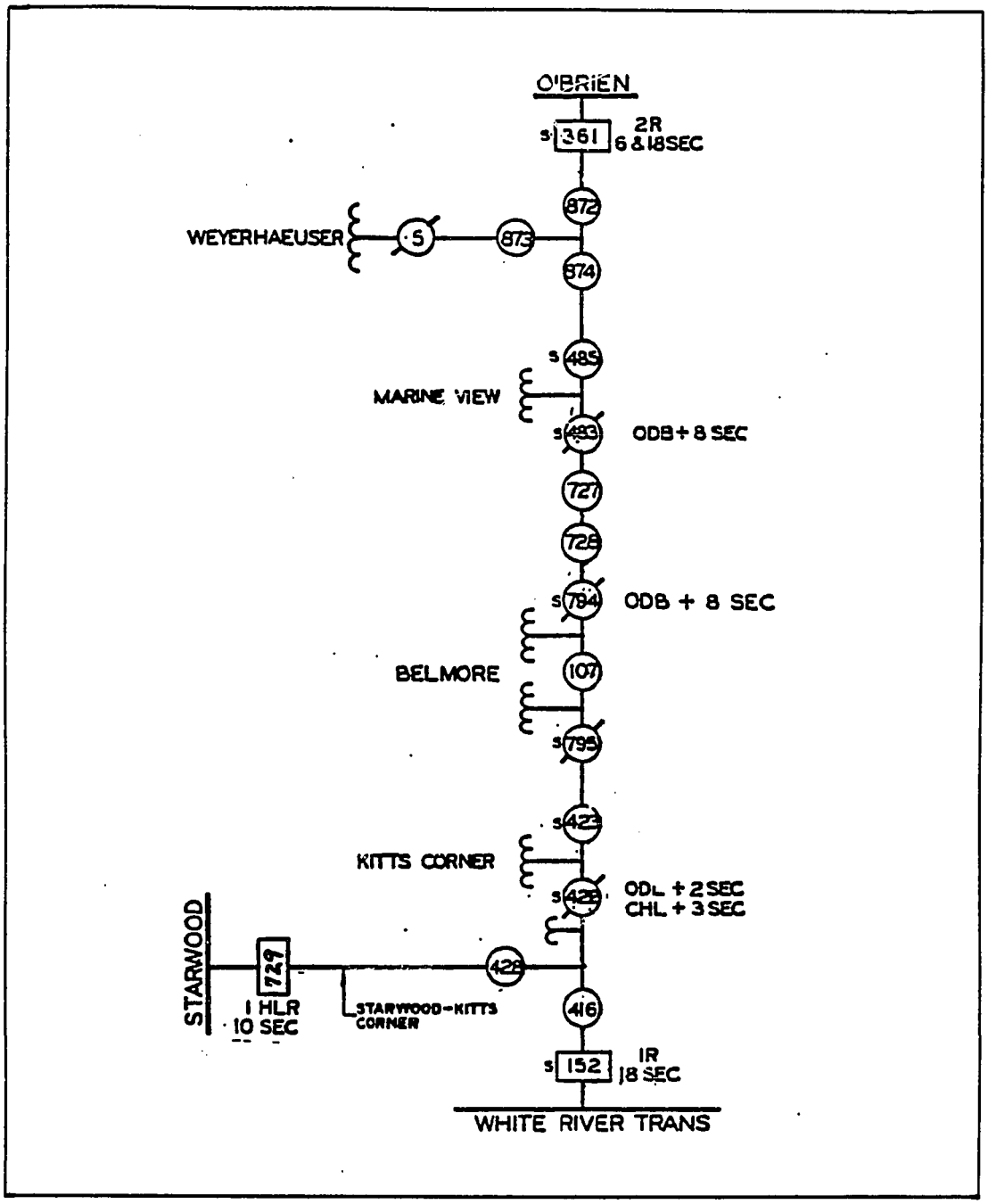


Fig. 2-9 O'Brien-White River Line for Cases 1-2

Close load break 794

Close load break 483

Status- Fault isolated between switches 874 and 485.

Case 2 (Fig. 2-9)

Faulted Section: The line section between switches 727 and 728.

Final configuration of protection devices:

Load break 483 open

Load break 794 open

All other devices are closed

Final solution: Fault is isolated between load breaks 794 and 483. All customers have service. Note that in this case switches 727 and 728 are neither automatic nor supervised. Since no customers are affected by the fault, no further test action will be suggested.

Case 3 (Fig. 2-10)

Faulted Section: The line section between switches 287 and 866.

Final configuration of protection devices:

Switch 618 remains open

Load break 944 open

All other devices remain closed

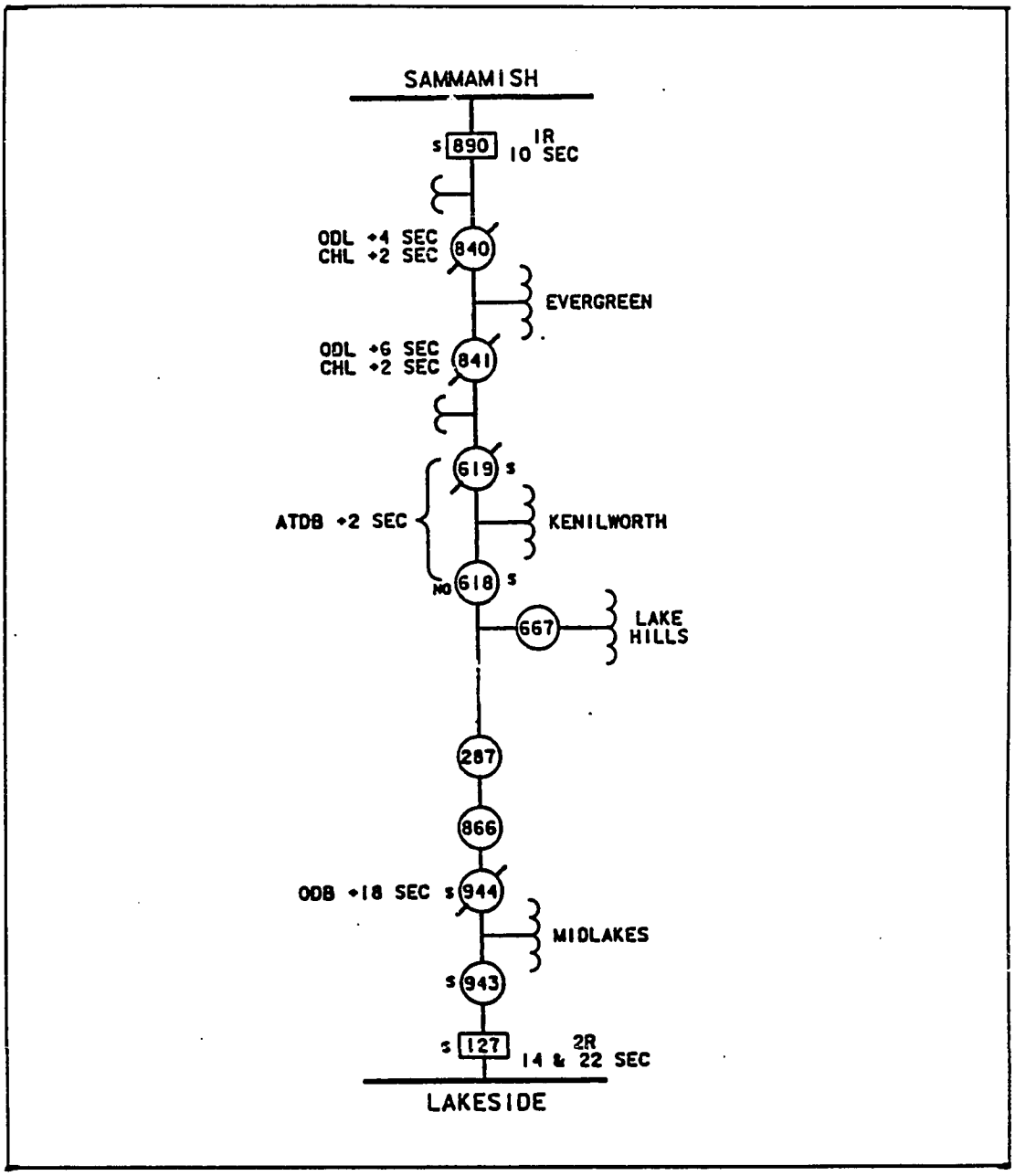


Fig. 2-10 Sammamish-Lakeside Line for Cases 3-4

Testing:

None- no testing can be performed without risking power loss to customers with service.

Final solution: Send field personnel to inspect line between switch 618 and load break 944. Power is out at Lake Hills; other substations will be served when all automatic switching actions are complete.

Case 4 (Fig. 2-10)

Faulted Section:

Same as Case 3.

System Condition: High wind. Several line outages.

Final configuration of protection devices:

Same as Case 3.

Testing:

Open section switch 287

Close section switch 618

Test successful

Final solution: Fault is isolated between section switch 287 and load break 944. All customers have service. The severe weather condition results in a test action whose failure will cause interruption of Evergreen and Kenilworth substations.

2.5 Dispatchers' Evaluation

The expert system was demonstrated to PSPL dispatchers a number of times in order to evaluate the system performance. Many of the dispatchers' comments related to the man/machine interface, for example using familiar terminology and

concise messages. In all tested scenarios, the expert system suggested a correct solution. Comments from these demonstrations resulted in adding the ability to handle multiple lines at the same time. Dispatchers use their judgment as to which line they want to work on first. This extension was deemed necessary since the expert system would be most helpful during abnormal weather conditions where several faulted lines may be expected.

In general, dispatchers are quite enthusiastic about the developed expert system and see several potential advantages for themselves and customers. It was felt that the developed system could reduce tedious and redundant manual tasks and therefore should lead to efficient restoration in emergency conditions. An immediate result is that the expert system helps minimize customer outage duration. Even in situations where the problem is simple, the expert system is useful in that it serves as a double check for their decision. On the actual implementation in control centers, however, a primary concern of dispatchers is to limit the amount of work they must perform to maintain system data by inputting untelemetered system information. The reason is that if they must spend a significant amount of time entering data in emergency situations, the usefulness of the proposed system will be greatly reduced.

2.6 Further Work

CRAFT is currently undergoing implementation in the PSPL control center. Implementation has generated ideas for enhancements. To begin, although scenarios which satisfy the initial assumptions in Sec. 2.2 cover the most common situations, these assumptions are fairly restrictive. One situation to consider is that of incorrect data. In this case, other information, perhaps analog measurements, may be useful in determining what is wrong. Research in this area is on-going at the University of Washington. Consideration of bus faults appears to be relatively straightforward and may be addressed in the future. The occurrence of more that one fault in a line is extremely rare and probably will not be considered. However, loosening some of the other assumptions may prove useful in the future. Finally, for this to be a useful dispatchers' tool, improvements in the user interface will be needed.

3. EXPERT SYSTEM APPLICATION TO REACTIVE POWER/VOLTAGE CONTROL

VCES was developed to assist decision-making for voltage problems in the power system. The problem of reactive power/voltage management was chosen because heuristic rules exist in the area. For example, if a bus voltage is low, the operator can switch on a shunt capacitor or raise the transformer tap position at that bus to restore the voltage to a normal value. Also, the area of reactive management has been studied extensively in the past [24]. Therefore systematic methods are available for the solution of the problem. For example, a reactive security dispatch algorithm is developed using the linear programming method [16]. A thorough methodology is proposed for the reactive power/voltage control under normal and emergency conditions [12]. Basically, if a bus voltage violates the normal operating limit slightly, then there is a very good possibility that heuristic rules can be applied to correct the problem. If, however, some bus voltage changes far beyond the normal operating limits or the voltage problem is widespread, then it should be more efficient to apply a systematic method, such as those presented in [16,24], to obtain a solution. Based on the above ideas, an ES approach is suggested for the reactive management. The proposed

system is capable of performing the detection of voltage problems and the application of empirical knowledge to come up with appropriate control actions. Furthermore, when it is judged that a systematic method is necessitated for an emergency condition, the expert system is designed to assist the user in formulating the problem so that a software package can be effectively utilized. Empirical rules for reactive power voltage control via shunt capacitors, transformer tap changer and generator voltages are identified. These rules are justified theoretically [12]. Based on the identified knowledge, a set of production rules is proposed. Sensitivities of various controls are incorporated in the developed rules to enhance the capability of the ES.

The organization of this chapter is as follows. The power system model is given in Sec. 3.1. The original proposed expert system for reactive power/voltage management, including the system configuration, the knowledge base, the data structure and production rules, is described and discussed in Sec. 3.2-3.3. In Sec. 3.4, extensions and results of numerical testing on the updated VCES are summarized. Details of extensions and a performance analysis based on extensive simulations follow.

3.1 Power System Model

The mathematical model of the power system will be

presented in this section. Consider a power system with n buses (substations). Load buses will be numbered from 1 to n_L and generator buses from n_L+1 to n (= $n_L + n_G$). For each bus i, there are four network variables: the complex voltage $V_i\underline{/\theta_i}$, and the real and reactive power injections P_i, Q_i, respectively. Transmission lines are modeled by the π-equivalent, i.e., a series impedance with shunt capacitors on both sides. In the steady-state, generations are balanced by load demands and line losses. Under this condition, the power system can be described by the well-known power flow (load flow) equations.

$$f_i(\underline{\alpha},\underline{V}) = \sum_{k=1}^{n} (V_iV_kB_{ik}\sin(\alpha_i-\alpha_k) + V_iV_kG_{ik}\cos(\alpha_i-\alpha_k)) = P_i \qquad (3.1)$$

$$g_i(\underline{\alpha},\underline{V}) = \sum_{k=1}^{n} (V_iV_kG_{ik}\sin(\alpha_i-\alpha_k) + V_iV_kB_{ik}\cos(\alpha_i-\alpha_k)) = Q_i \qquad (3.2)$$

where $\underline{V} = [V_1,V_2,\cdots,V_n]^T$, $\underline{\alpha} = [\alpha_1,\alpha_2,\cdots,\alpha_{n-1}]^T$ and $\alpha_i = \theta_i - \theta_n$ (bus n is chosen as the slack bus), and B_{ij}, G_{ij}, are the line susceptance, conductance of the line from bus i to j, respectively.

The load model is now discussed. To meet different loading conditions during the operation, transformer tap changers can be adjusted to maintain the bus voltage. At a load bus i with a transformer tap changer, the primary and secondary voltages V_i, $\bar{V}_i$, respectively, and the tap

setting t_i, are related by $\bar{V}_i = t_i V_i$. In general, real and reactive power demands are voltage dependent. Therefore, the injections at a load bus are assumed to be of the form

$$P_i = P_{io} \bar{V}_i^{a_i} \tag{3.3}$$

$$Q_i = Q_{io} \bar{V}_i^{b_i} \tag{3.4}$$

where the constants P_{io}, Q_{io}, a_i, b_i can be determined from the load characteristics at bus i. A reported survey of load characteristics has shown that typical ranges for the exponents a_i and b_i are 0.7-1.2, 1.0-2.0, respectively. For the development here, $a_i, b_i > 0$ is the only necessary assumption. To solve the load flow equation, normally the quantities P_i, V_i at a generator bus and P_i, Q_i at a load bus are specified. A slack bus with constant V_i, θ_i is also designated. In the proposed model, however, the load is voltage dependent. Therefore, the injections P_i, Q_i are given by Eqs. (3.3),(3.4) instead of specified constant values. With the proposed load model and the specified quantities at generator buses, the power flow equations (3.1-3.2) can be solved to obtain the operating point $(\alpha_1,\cdots,\alpha_{n-1},\bar{V}_1,\cdots,\bar{V}_{n_L})$. The operating point of a power system is subject to security constraints, including bus voltage limits, generator and line capacities, etc. The voltage limits are denoted by a set of inequalities

$$\bar{V}_i^m \le \bar{V}_i \le \bar{V}_i^M \qquad i=1,\cdots,n_L \tag{3.5}$$

$$V_j^m \le V_j \le V_j^M \qquad j=n_L+1,\cdots,n \tag{3.6}$$

Under normal (emergency) conditions, load bus voltage limits are, typically, $\overline{V}^m$ = 0.95 (0.90) p.u. and $\overline{V}^M$ = 1.05 (1.10) p.u. Generator bus voltages are controlled quantities and the range of variation in Eq. (3.6) may be smaller, depending on generator reactive power limits. When the load voltages violate the constraints of Eq. (3.5), accepted practice is to employ reactive compensators, such as switched shunt capacitors, reactors, transformer tap changers and synchronous condensers to restore the voltage profile. Generator bus voltages can also be adjusted to maintain the load voltages. In this study, reactive compensators are modeled by reactive injections Q_i, and transformer tap settings t_i. These controls are subject to the following constraints.

$$Q_i^m \le Q_i \le Q_i^M \qquad i=1,2,\cdots,n_L \qquad (3.7)$$

$$t_i^m \le t_i \le t_i^M \qquad (3.8)$$

where Q_i^m, t_i^m and Q_i^M, t_i^M are, respectively, the lower and upper limits of the controls.

This study is concerned with the operation of reactive control devices to maintain a normal voltage profile. If the utilization of the above controls fails to restore the voltage profile, disruptive control techniques such as load shedding will be needed. Disruptive controls have not been considered in this study.

3.2 Knowledge Acquisition Results

A successful ES relies on a high-quality knowledge base. The knowledge base needs to incorporate the knowledge (facts, heuristics) required to perform a task well. For the reactive power/voltage control problem, the following empirical rules are identified:

(a) If a load bus voltage drops below (or rises above) the operating limit, control devices such as shunt capacitors, transformer tap changers, generator voltages and synchronous condensers can be switched or adjusted to restore the voltage profile.

(b) If a load bus voltage drops below (or rises above) the operating limit, it is most efficient to apply the reactive compensation locally. If the capacity of local compensators is insufficient to resolve the voltage problem, then the compensators with the next highest sensitivities should be chosen.

(c) If the voltage is low (high) at a bus, the tap position of the local transformer tap changer can be raised (lowered) to correct the problem. However, increasing (decreasing) the tap position may cause other load bus voltages to drop (rise).

(d) Generator bus voltages can be raised (lowered) to solve the low (high) load voltage problems.

In the steady-state security assessment framework

developed by DyLiacco [25], a system is said to be in an emergency state if any operating limit is violated. Intuitively, if a slight violation occurs, one should be able to obtain appropriate control actions based on empirical rules (b),(c),(d). If, however, the violation is excessive, e.g., a very low bus voltage, or the voltage problem is widespread, then an available software package will be necessary to generate the control actions systematically. Based on different levels of violations, we propose the following classification which will be denoted by a severity index S:

S	Operating Condition	Description of the problem
0	Normal	All voltage limits satisfied.
1	Emergency	Slight violation of the voltage limit at one or two buses.
2	Emergency	Severe violation of one voltage limit or three or more bus voltage limits violated, and local and neighboring control devices are sufficient to solve the problem.
3	Emergency	Same as above, except local and neighboring control devices are not sufficient to solve the problem.

Based on the above classification, the empirical rules (a)-(d) apply for problems where S=1. The following heuristic rules are obtained for S=2 or 3.

(e) If the index for the system voltage condition S=2 or 3, a more systematic method such as a reactive power dispatch algorithm using linear programming is required to efficiently find a solution.

(f) If the index for the system voltage condition S=2, the objective of the reactive power dispatch algorithm should be minimize the number of control devices involved.

(g) If the index for the system voltage condition S=3, the reactive power dispatch algorithm should be utilized to find a feasible solution as quickly as possible. Other objectives have lower priorities a this time. If necessary, emergency bus voltage limits should be used.

Among the above rules (a)-(g), (a) is well accepted. Theoretically (b), (c), and (d) can be established based on systematic analysis [12]. Finally, rules (e),(f),(g) involve the classification of different levels of emergency conditions and the choice of the objective function for the optimization problems. Hence, some modification may be necessary, depending on an individual's judgments. The

proposed empirical rules represent the knowledge we have acquired from the literature and discussions with engineers in the field.

3.3 Expert System Development

In this section, an expert system which is developed to aid the user in detecting voltage problems and determining appropriate control actions is presented. The ES configuration is discussed in Sec. 3.3.1. Sec. 3.3.2 deals with the data structure of the ES implemented in OPS5. Based on the identified knowledge, a set of production rules is proposed in Sec. 3.3.3.

3.3.1 The ES Configuration

An ES consists of three fundamental components: a global data base, a knowledge base, usually a set of IF-THEN structures which represent the knowledge required to solve a problem, and an inference engine utilized to chain a set of rules to form a line of reasoning. The knowledge used by human experts in solving a problem may include facts derived from physical laws and heuristics based on engineering judgments. To solve a complex problem, the knowledge required may be sophisticated which often results in a large rule base. In this case, an efficient inference procedure needs to be developed to search for the applicable rules.

In this study, the proposed objective of the ES is to aid

in the detection of voltage problems and the search for a control scheme to correct the detected problem. As shown in Fig. 3-1, the proposed rules are separated into two sets depending on the functions they perform. Rule Base 1 performs the detection of voltage problems and the search for control actions based on empirical knowledge. When it is judged that the problem is serious enough that the empirical judgment may not be reliable, Rule Base 2 aids the user in formulating the problem in order to utilize an available LP which provides a more systematic method. Therefore, the role of the proposed expert system is also to assist the user in the utilization of the available software packages.

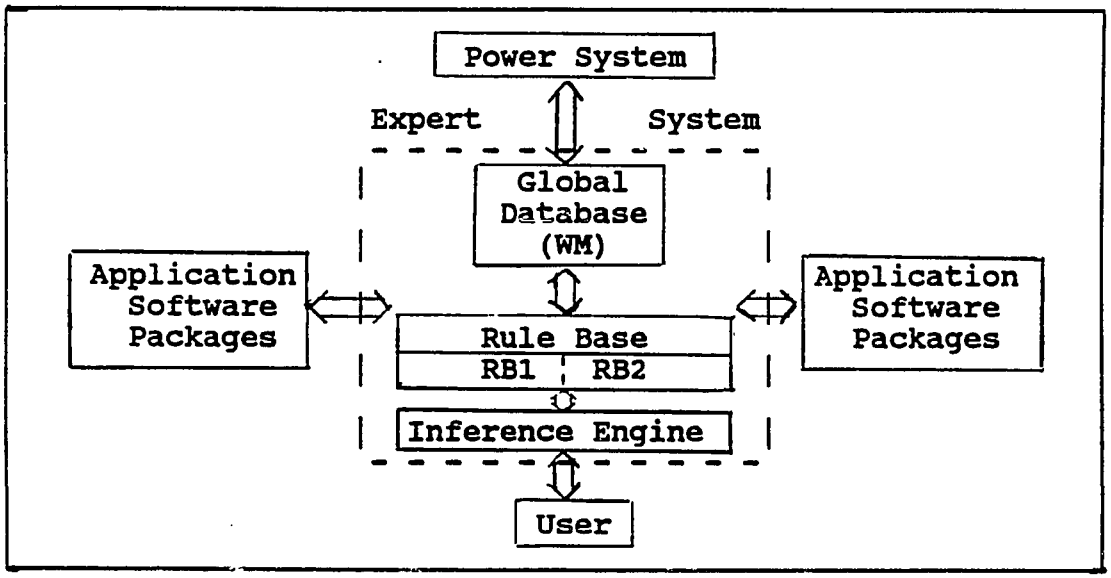


Fig. 3-1 The Expert System Configuration

3.3.2 The Data Structure

The general form of the data representation is given by

type, name, status, attributes, characteristics. The above form is explained explicitly below for various data types.

(1) Load Buses

(name,location,status,MW demand, MVAR demand, voltage V, V-upper limit, V-lower limit)

(2) Reactive controllers (RC) -- tap settings, reactive injections, generator voltages

(name, type, location, status, current value, upper limit, lower limit)

3.3.3 Production Rules

For the reactive power management, we developed 28 rules based on the knowledge described in Section 3.2. RB1 concerns itself with less severe voltage problems, attempting to correct the voltage problem with a small amount of computation by utilizing the first 12 rules listed below. Sensitivities for reactive compensators, generator voltage control, and transformer tap changers are incorporated into the RB1 rules to obtain best estimates of the control required to restore the voltage profile. RB2 formulates decisions for more severe voltage problems by utilizing a linear programming package similar to [16]. In these rules, sensitivities and available control capacities are used to classify the severity of the voltage problem. Voltage problems of severity rating 3 are solved using emergency constraints with the cost function for the LP

package set to zero. Severity rating 2 problems are solved with normal constraints and the objective is to minimize the weighted sum of the control adjustments.

Before presenting the condensed version of the developed rules, a complete description of rule 1 is given.

Rule1-- If a load bus voltage V_i is below the normal voltage limit and no voltage violations have been detected then identify bus i as a low voltage bus and compute the voltage deviation 1.0 (p.u.) - V_i.

A realization of the above rule in OPS5 follows.

```
(p 1
(pq_bus ^name <bus> ^v_lower <lower> ^voltage {>=0.90 <
<lower> <volts>})
-(severity)
-(voltage_controller)
-->
(bind <delV> (compute 1.0 - <volts>))
(make voltage_controller ^delV <delV> ^type low
      ^violator <bus>)
(write (crlf) voltage at <bus> is <volts> -- (crlf)
        violates lower voltage constraint))
```

The condensed RB1 production rules are now presented.

Rule 1--If a load bus voltage is below voltage limits and above emergency voltage limits then identify bus as having a low voltage.

Rule 2--If a load bus voltage is above voltage limits and below emergency voltage limits then identify bus as having a high voltage.

Rule 3--If a problem bus voltage is now within limits then clear voltage control and verify with load flow.

Rule 4--If the voltage violation is a low voltage and the compensator to be used for control is available then compute maximum voltage increase

$$\Delta V_i = d_{ij}\Delta U_j$$

where d_{ij} is the sensitivity defined in [6] for the various controllers, ΔV_i is the maximum voltage increase and ΔU_j is the available control.

Rule 5--If the voltage violation is a high voltage and the compensator to be used for control is available then compute maximum voltage decrease (see Eq. in Rule 4).

Rule 6--If the compensator can provide the necessary control then implement the necessary compensation.

Rule 7--If control compensator is not at limits and compensation is insufficient to eliminate voltage violation then move compensator to limit.

Rule 8--If the compensator to be used for control has not been chosen then identify voltage controller.

Rule 9--If the control compensator is at full output and there remain controllers to be checked then select next controller.

Rule 10--If the control compensator is not available and there are no more controllers to be checked then severity rating=3.

The final sixteen rules, RB2, are given in condensed form below.

Rule a1--If more than two load bus voltages are below voltage limits then initiate determination of problem severity.

Rule a2--If more than two load bus voltages are above voltage limits then initiate determination of problem severity.

Rule a3--If both high and low voltage violations are present then severity rating is 3.

Rule a4--If a load bus voltage is outside voltage emergency limits then initiate determination of problem severity.

Rule a5--If the n most significant (sensitive) controllers have been determined to be insufficient to alleviate the voltage problems then severity rating=3.

Rule a6--If the n most significant controllers have been determined to be sufficient to alleviate the voltage problems then severity rating=2.

Rule a7--If a load bus voltage is below voltage limits and has not been checked for available control then determine magnitude of voltage violation.

Rule a8--If a load bus voltage is above voltage limits and has not been checked for available control then determine magnitude of voltage violation.

Rule a9--If a low voltage exists and the controller under consideration is available then calculate voltage control available from this controller (see Eq. in Rule 4).

Rule a10--If a high voltage exists and the controller under consideration is available then calculate voltage control available from this controller (see Eq. in Rule 4).

Rule a11--If the first controller on the list of controllers to be checked is unavailable then select next controller.

Rule a12--If severity rating=2 then run LP problem using normal constraints and unity weighting factor for all control adjustments.

Rule a13--If severity rating=3 then run LP problem using emergency constraints and feasibility only (set cost function to zero).

Rule a14--If LP cannot resolve the voltage violations using normal constraints then increase the severity rating to 3.

Rule a15--If LP cannot resolve the voltage violations using emergency constraints then inform operator that system cannot be kept within emergency limits using non-disruptive controls.

Rule a16--If all voltages are within voltage emergency limits then clear voltage emergency.

3.4 Simulation Results

A modified IEEE 30-bus system (Fig. 3-2) is chosen to demonstrate the usefulness of the expert system structure and proposed rules. The system parameters can be found in [26] with the following changes:

(1) Reactive Compensators (RCs) are available at buses 10, 19, 24, and 29.

(2) Transformer tap changers are available at buses 4, 14, 16, and 26.

(3) Load characteristics for all loads are assumed to be

$$Q_i = Q_{io}\bar{V}_i^{1.2}$$

$$P_i = P_{io}\bar{V}_i^{1.2}$$

where P_{io} and Q_{io} are the constant load demands given in [27]. For each example, modifications are made in the loading condition in order to create scenarios of voltage problems. The resulting rule firings within the expert system for these scenarios is then given. Table 3-1 presents initial system conditions and the resulting conditions from applying the control plan proposed by the expert system.

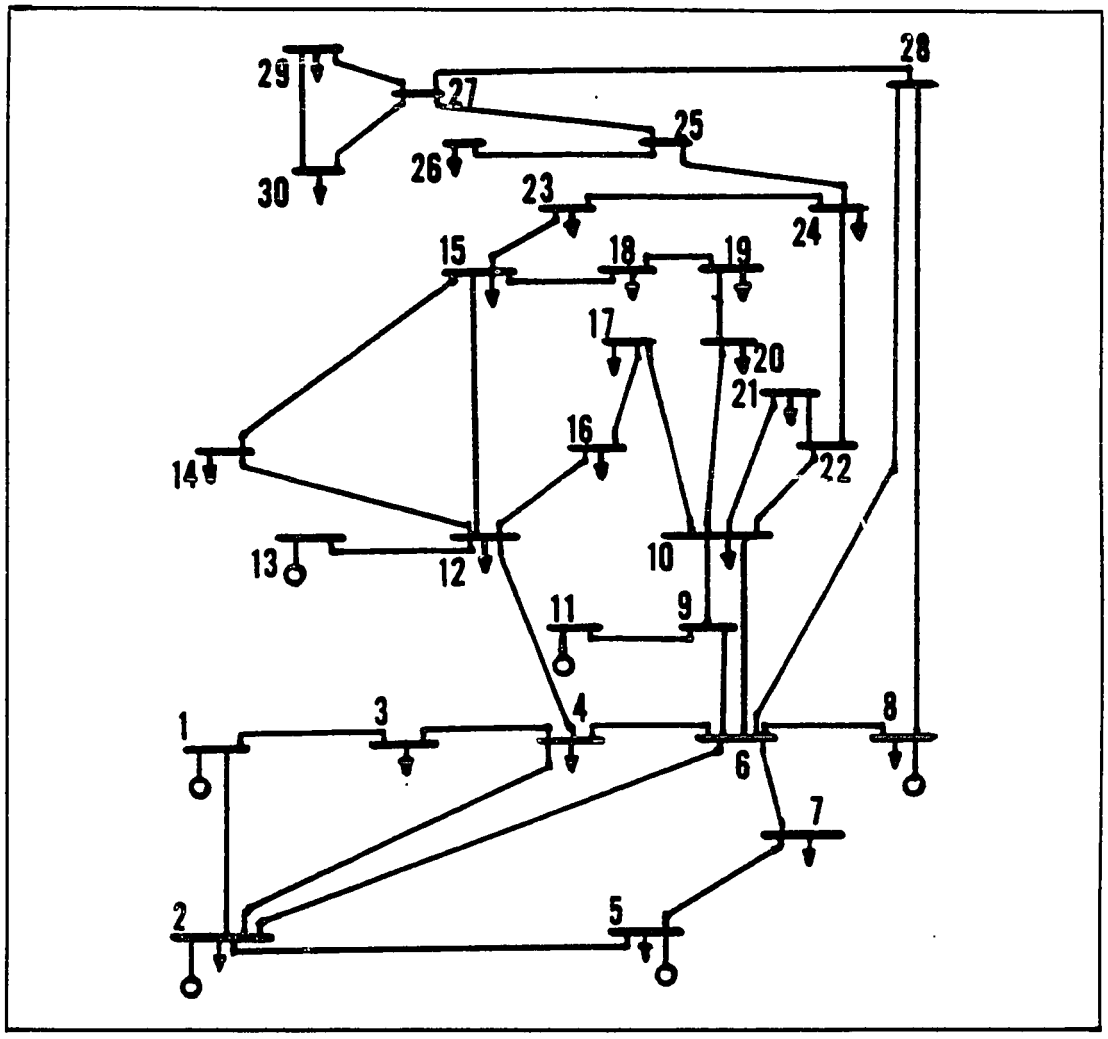


Fig. 3-2 IEEE 30-Bus System

Case 1

Problem: Low voltages (<0.95) at bus 26, V = 0.946 and bus 30, V = 0.942

Solution: Reactive compensation at bus 29 is applied to solve the voltage problem.

Sequence of applying rules: Rule 1: Voltage at bus 30 low; Rule 8: Choose RC at bus 29 as controller; Rule 4:

Compute available voltage increase; Rule 7: RC 29 moved to upper limit; Rule 3: Voltage within limits

An extension to this problem was considered by assuming that reactive compensation is unavailable at buses 24 and 29. The sequence of applying rules is similar to above with the control decision consisting of an attempt to increase the generator voltage at bus 8 to 1.10 . However, generator reactive limits are reached. Therefore the voltage problem is alleviated with a maximal generator voltage of 1.07.

Case 2

Problem: Low voltage at bus 26, V = 0.946 (<0.95). In this example, RCs at bus 29 and bus 24 are assumed to be at full output. and Generator voltage limits at bus 8 and bus 11 are lowered to 1.03 p.u. so that these controls are no longer available.

Solution: Tap setting at bus 26 is raised to solve the problem.

Sequence of applying rules Rule 1: Voltage at bus 26 low; Rule 8: Choose RC at bus 29 as controller; Rule 4: Compute available voltage increase; Rule 9: RC 29 moved to upper limit; Rule 4: Compute available voltage increase ; Rule 9: RC 24 moved to upper limit; Rule 4: Compute available voltage increase; Rule 9: Generator 8 moved to upper limit; Rule 4: Compute available voltage increase; Rule 9: Generator 11 moved to upper limit; Rule 4: Compute available voltage increase; Rule 6: Tap 26 is sufficient; Rule 3:

Voltage within limits.

Case 3

Problem: Low voltage (<0.95) at bus 25, V = 0.940 bus 26, V = 0.899 bus 27, V = 0.949 bus 29, V = 0.910 bus 30, V = 0.892

Solution: Adjustment of the tap at bus 26, increased reactive injection at buses 19,24, and 29, and the generator voltage at bus 1 is raised as well in order to alleviate the low voltage problem.

Sequence of applying rules Rule a1: More than 2 low voltages; Rules a7,a9,a11: Repeated to compute available control; Rule a6: The severity index s=2; Rule a12 : Run LP with unity weighting factors and normal constraints, voltage problem solved.

An extension to this problem was considered by assuming the following limits on the controllers: RC 24 is unavailable, RC 29 has 0.2 MVAR available for control, Generator voltage at bus 8 is at maximal value. The sequence of applying rules is the same with the exception of the last two rules. Rule a5 : The severity index s=3 ; Rule a13 : Run LP with emergency constraints and zero cost function, voltage problem is solved with emergency constraints.

Table 3-1 Results of Cases 1,2, 3

Variables		Limits		Case 1		Case 2		Case 3	
		Low	High	Init.	Final	Init.	Final	Init.	Final
RCs (MVAR)	Q_{10}	-1.0	1.0	0	0	0	0	0	0
	Q_{19}	-1.0	1.0	0	0	0	0	0	0.30
	Q_{24}	-1.0	1.0	0	0	1.0	1.0	0	1.0
	Q_{29}	-1.0	1.0	0	1.0	1.0	1.0	0	1.0
Gen. Volt.	V_1	1.00	1.10	1.06	1.06	1.06	1.06	1.06	1.062
	V_2	1.00	1.10	1.045	1.045	1.045	1.045	1.045	1.045
	V_5	1.00	1.08	1.01	1.01	1.03	1.03	1.01	1.01
	V_8	1.00	1.08	1.01	1.01	1.03	1.03	1.01	1.01
	V_{11}	1.00	1.10	1.08	1.08	1.08	1.08	1.08	1.08
	V_{13}	1.00	1.10	1.08	1.08	1.08	1.08	1.08	1.08
On-load Taps	t_4	0.90	1.10	1.0	1.0	1.0	1.0	1.0	1.0
	t_{14}	0.90	1.10	1.0	1.0	1.0	1.0	1.0	1.0
	t_{16}	0.90	1.10	1.0	1.0	1.0	1.0	1.0	1.0
	t_{26}	0.90	1.10	1.0	1.0	1.0	1.06	1.0	1.032
Load Volt.	V_3	0.95	1.05	1.026	1.026	1.033	1.033	1.024	1.026
	V_4	0.95	1.05	1.018	1.019	1.027	1.027	1.016	1.018
	V_6	0.95	1.05	1.012	1.013	1.025	1.025	1.010	1.012
	V_7	0.95	1.05	1.003	1.004	1.019	1.019	1.002	1.004
	V_9	0.95	1.05	1.024	1.026	1.032	1.031	1.020	1.028
	V_{10}	0.95	1.05	1.002	1.005	1.011	1.010	0.996	1.010
	V_{12}	0.95	1.05	1.031	1.032	1.036	1.035	1.027	1.033
	V_{14}	0.95	1.05	1.012	1.015	1.018	1.018	1.007	1.015
	V_{15}	0.95	1.05	1.006	1.009	1.012	1.012	0.998	1.008
	V_{16}	0.95	1.05	1.011	1.013	1.018	1.017	1.006	1.016
	V_{17}	0.95	1.05	0.999	1.002	1.007	1.007	0.994	1.006
	V_{18}	0.95	1.05	0.992	0.995	0.999	0.993	0.984	0.997
	V_{19}	0.95	1.05	0.987	0.990	0.995	0.994	0.980	0.993
	V_{20}	0.95	1.05	0.989	0.993	0.998	0.997	0.981	0.998
	V_{21}	0.95	1.05	0.989	0.994	0.993	0.998	0.981	0.999
	V_{22}	0.95	1.05	0.989	0.994	0.998	0.998	0.981	0.999
	V_{23}	0.95	1.05	0.989	0.994	0.997	0.997	0.972	0.991
	V_{24}	0.95	1.05	0.975	0.984	0.986	0.985	0.959	0.990
	V_{25}	0.95	1.05	0.971	0.992	0.983	0.981	0.940	0.975
	V_{26}	0.95	1.05	0.946	0.968	0.946	1.000	0.899	0.962
	V_{27}	0.95	1.05	0.980	1.009	1.001	0.999	0.949	0.943
	V_{28}	0.95	1.05	1.007	1.011	1.022	1.022	1.002	1.008
	V_{29}	0.95	1.05	0.961	1.020	0.992	0.991	0.901	0.980
	V_{30}	0.95	1.05	0.942	0.988	0.976	0.974	0.892	0.948

Remark Note that in all cases the response of bus voltages to various controls agrees perfectly with results of Facts(1-3) in Sec. 3.2.

3.5 Performance Evaluation

In this section, the capability and computational efficiency of VCES are compared with results of an LP for reactive dispatch, see [17]. Different scenarios of a test system are selected as a basis for comparison. For the expert system approach, the computation time is further broken down to basic operations of forward-chaining execution, i.e. attribute testing, intercondition testing, conflict resolution, actions, and various overhead times. The cost of executing any individual one of these operations (e.g., testing one attribute) is basic because their computational costs are measurable and relatively insensitive to the particular problem of application. From an application point-of-view, the breakdown allows one to identify the bottleneck of computation of an ES. The results are useful in evaluating the quality of an ES and improving efficiency of the rule-based program.

In the process of development, several extensions [17] were made beyond the original work of [12]. First, the knowledge base is extended to allow VCES to deal with wider-spread voltage problems. Reactive resources are compared with low voltages in an area; sensitivity factors are used

to determine if all low voltages in the area can be eliminated at the same time. Second, VCES incorporates discrete constraints on control settings. Third, the rule-based program VCES which was originally implemented in OPS5 is rewritten in OPS83. This results in higher computational efficiency. Furthermore, the new language greatly simplifies the interface with FORTRAN and therefore we are able to integrate the VCES with two FORTRAN programs: the load flow, and a reactive dispatch algorithm based on linear programming (LP) [16]. The ES, LP, and ES/LP methods are applied to solve a number of scenarios with varying severity of a test system. The three methods are compared in terms of the quality of solution and the computational speed. In this study, the expert system processing is broken down to a number of basic tasks, for which the computational costs are measured. Since the measurements are designed to be problem independent, they are used to estimate the processing costs of these tasks for the selected voltage problem scenarios.

The proposed rule-based system VCES serves as an operator's aid in resolving voltage problems. It was suggested in the last section that empirical rules be used when the problem is minor(few low voltages). On the other hand, if the problem is severe then a reactive dispatch algorithm will be called upon to generate a solution. The general framework of the integrated rule-based and number-crunching programs can be described by Fig. 3-3. It is noted

that to apply the linear program(reactive dispatch) it is necessary to linearize the system equations around the selected operating point. The interface block in Fig. 3-3 represents the solution of a load flow and evaluation of partial derivatives for use in the formulation of the linear program. Note that if the operating point is available from the system, the load flow solution would not be necessary. However, if it is desirable to verify the selected controls with the load flow before it is applied to the power system, then more than one LP solution may be needed. In this case, it is necessary to solve the load flow to update the parameters in the LP.

In Section 3.5.1, various components of the computational tasks in the ES will be broken down in a precise manner. Measurements of computational costs of the tasks will be used to evaluate the efficiency of VCES; the results are provided in Section 3.5.2. An analysis of these results follows.

3.5.1 Efficiency Issues - Initial Considerations

The execution of an expert system is different from that in a procedural language such as FORTRAN. The diagram in Fig. 3-4 illustrates how a forward-chaining rule-based

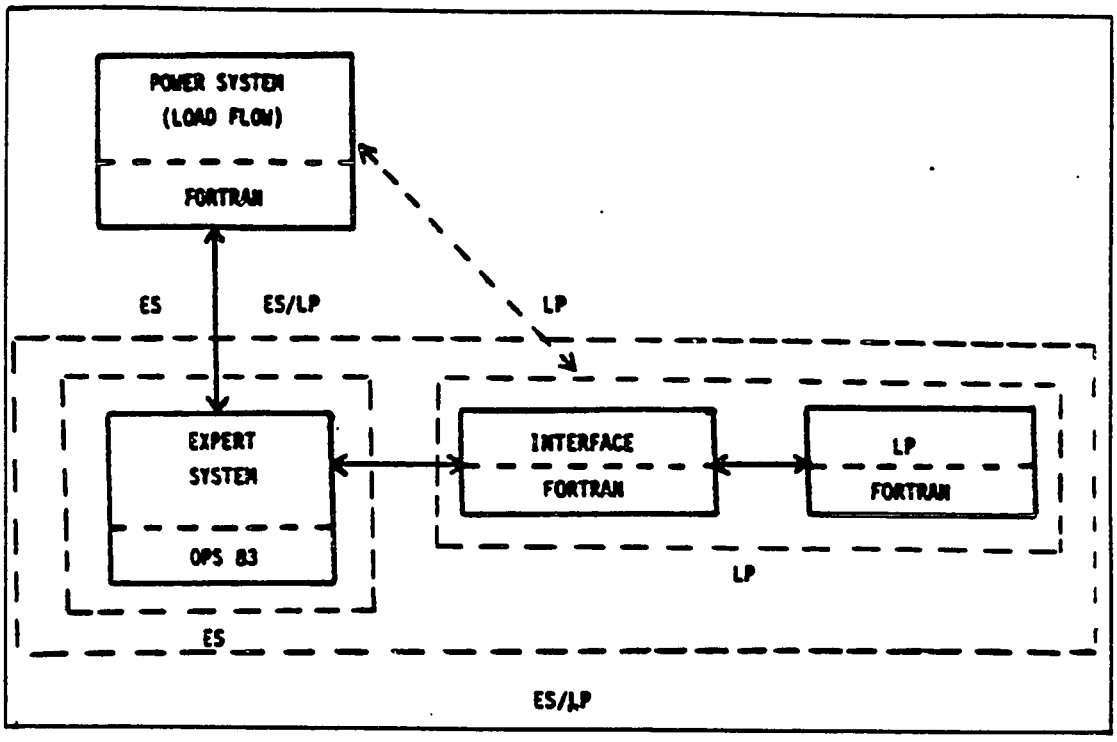


Fig. 3-3 Voltage Control Expert System(VCES)

program is executed. There are three major steps: rule-search, conflict resolution, and act (rule-firing). The system operating condition is stored in the working memory(WM), organized by a certain data structure. For the system condition represented by the patterns in WM, the inference engine would perform the pattern matching to identify the set of rules that are satisfied by WM. The matched rule set needs to be searched according to a priority embedded in the inference engine. The most applicable rule will be fired. In other words, the action (conclusion) part of the rule will be executed.

Normally the action may initiate some changes in the data set and therefore the Pattern Matching-Conflict Resolution-Act cycle would have to be repeated. The iteration stops when no more rules are satisfied by any working memory elements or an explicit halt is encountered in the rule conclusion.

Expert systems may be quite slow since the entire rule base must be searched before each rule firing. It is important to identify those aspects which will slow down the program execution [28,29]. Due to the complexity of large rule-based systems it is difficult to obtain an exact

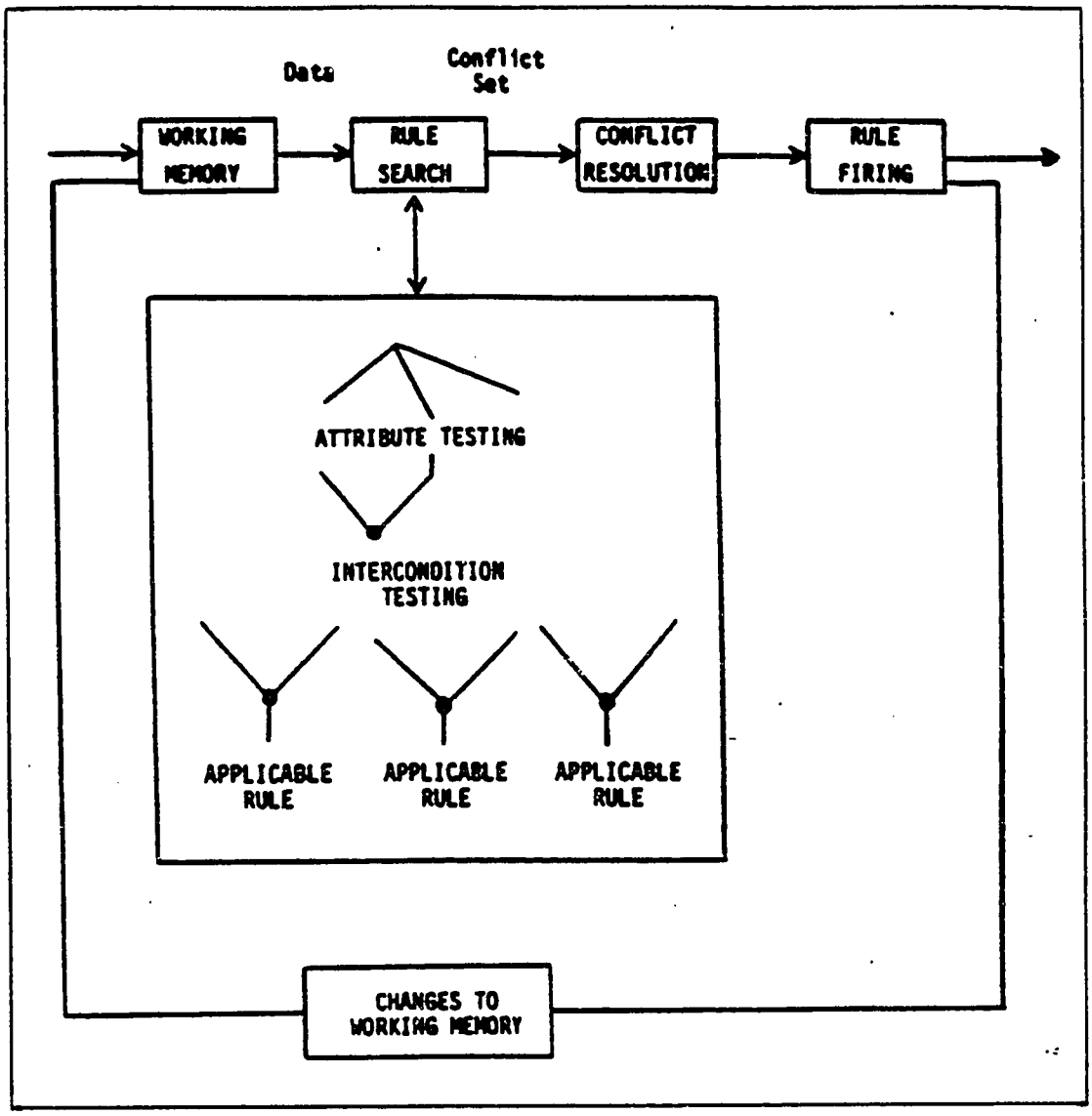


Fig. 3-4 Forward Chaining Mechanism in OPS83

expression for the processing that must be performed during execution. In this section, measurements on operations carried out within the inference engine when VCES is executed are used as guidelines for analyzing expert system performance.

OPS83 (and its predecessor OPS5) relies on a fast pattern matching algorithm called RETE [30]. The algorithm takes advantage of two characteristics of production systems in order to gain efficiency. First, many rules in a production system may have identical conditions. When this occurs, RETE shares the condition checking in order to avoid repeating the same operation. This is accomplished by compiling the rules into a network, called a discrimination net. Second, working memory tends to change slowly. Thus, RETE will store the results of any condition check and process only changes to working memory.

Shown below is a simplified rule from the developed VCES and the resulting discrimination net generated by OPS83 (Fig. 3-5). This rule applies when the highest priority controller at a bus with a low voltage violation is available. Subsequent rules will determine the appropriate control level for this controller. The rule is presented first descriptively and then in OPS83.

Rule A-- IF the current task is to identify control actions AND a voltage violation has previously been detected AND the highest priority control for the violating bus is in

service THEN select this controller for adjustment

In OPS83:

```
&task (task id=control_actions; );
&vv   (voltage_violator );
&bus  (bus type=pq; name=&vv.name; );
&ctrl (controller name=&bus.control[1]; status=active; );
-->
modify &vv (controller=&ctrl.name;);
```

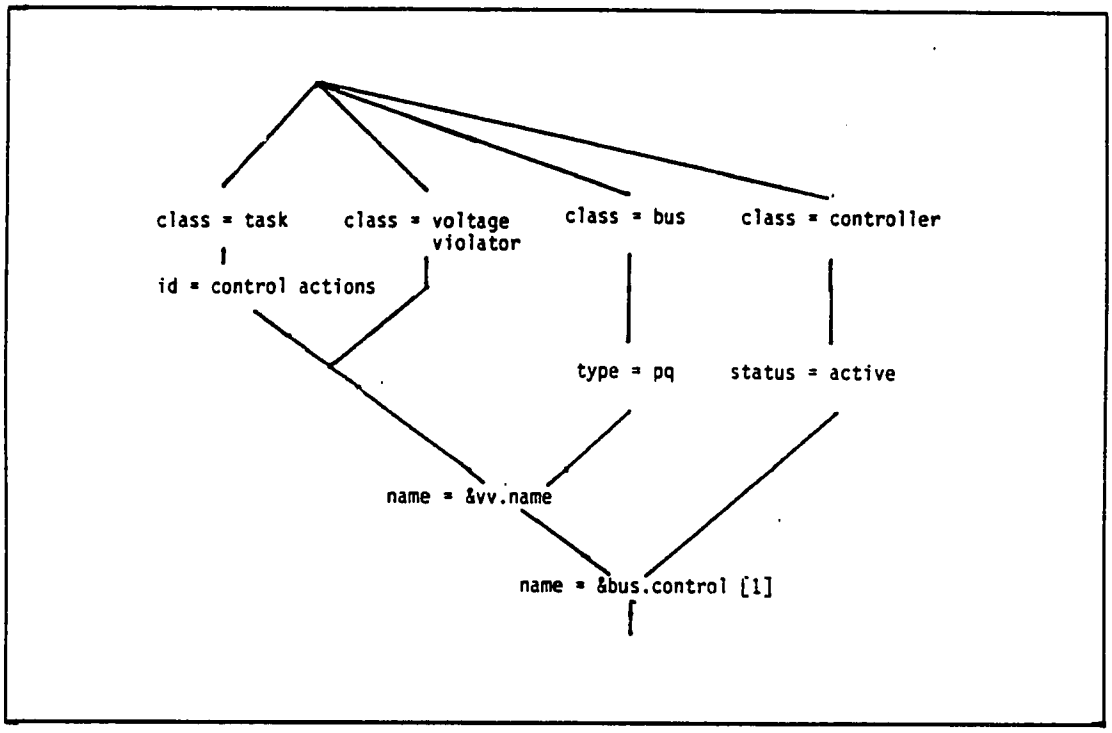


Fig. 3-5 Discrimination Net for Rule A

A number of operations must be performed in order for this rule to be satisfied. Individual conditions are dealt with independently. In this example, when the working memory elements task, voltage_violator, bus or controller are created, the corresponding branches in the discrimination net act to perform the necessary tests on their attributes

(e.g. name, id, etc.). We will refer to this processing as attribute testing. Now, if there are elements which satisfy conditions &task and &vv, then these elements must be combined and checked to determine if there are pairs of elements which satisfy these conditions. This combination of elements continues with the subsequent conditions (&bus and then &ctrl) until either there exists a set(or sets) of working memory which satisfies all conditions or the test fails. This processing between conditions is referred to as intercondition testing. For this example, the number of intercondition checks that must be performed is dominated by the number of previously detected voltage problems, the number of PQ buses which are violating their voltage constraints and the number of active controls within WM, i.e. the product of these 3 quantities.

Since several rules in VCES have similar pattern matching between voltage violators, low voltage PQ buses and active control sources, it is expected that the pattern matching time will grow exponentially with the number of voltage violations. In addition, computational time can be expected to increase as the length of the chain of reasoning increases, since more control decisions are required to resolve the violations.

Attribute & intercondition testing constitute the most time-consuming pattern matching aspects of the production system language. In addition, there are the times associated

with rule firing (overhead costs and any functions performed in the RHS) and the time required for conflict resolution. Conflict resolution is dependent on the size of the conflict set(i.e. the number of rules which are satisfied at any one step) and the nature of the resolution scheme.

There are a number of details associated with the operations described above [29]. In order to achieve meaningful measurements for ES developers, experiments are designed to be problem independent. For example, the attribute testing time is measured by a simple rule with an N-attribute condition. For very small N, say 1, the processing cost is mainly the overhead. As N increases, say to 100, one can exclude the overhead time and obtain a linear relationship between N and the total time for attribute testing. The measurements have been taken on a MicroVAX II computer and are presented in Table 3-2.

Table 3-2 Approximate Cost of Operations in OPS83 on MicroVax II.

Operation	Time(ms)
Attribute Testing	0.008/Attribute
Intercondition Testing	0.350/WM Element Pair
Conflict Resolution	0.130/Rule Pair
Changes to WM Overhead	0.430/Element
Rule Firing Overhead	0.385/Rule

3.5.2 Simulation Results

To compare the capability and efficiency of the ES, LP, and ES/LP methods. The standard IEEE-30 bus will be utilized [27]. The one-line diagram of the system is given in Fig. 3-2. Five cases with varying severity of the voltage problems, shown in Table 3-3, were selected for simulation. Note that the worst case, Case 5, contains 13 low voltages. A number of modifications were made to the IEEE-30 bus system beyond those given earlier, the complete modifications and the operating constraints are as follows:

1) Buses 10,19,24 and 29 have reactive compensation available with an upper limit of 0.10 MVAR in discrete steps of 0.01 MVAR.
2) On-load tap changers are available at buses 4,6,16 and 26 with an operating range of 0.90-1.10 in discrete steps of 0.01.
3) Generators are assumed to have an operating range of 1.00 p.u. to 1.10 p.u..
4) Normal operating voltage range for load buses is 0.95 p.u. to 1.05 p.u..
5) Load demands have been manipulated to create a variety of voltage profiles.

Table 3-3 - Test Scenarios

Case	# of Low Voltages	16	17	18	19	20	21	22	23	24	25	26	29	30
1	1													.944
2	2												.948	.927
3	5									.948	.939	.905	.922	.907
4	8	.943			.948	.949				.947	.940	.904	.928	.915
5	13	.941	.946	.943	.933	.933	.945	.946	.949	.935	.936	.906	.929	.915

For each scenario, the expert system and the LP are used to propose a solution for the operating condition. Additionally, the expert system in conjunction with the LP is used to propose a solution for Cases 4 and 5. In order to combine the ES and LP approaches, a threshold was set such that above a certain number of voltage violations, say 5, the LP was used. Below this threshold VCES will propose control actions. If the LP is required, the threshold is checked after each iteration, allowing VCES to make final control adjustments. In this section, these scenarios are presented and then a comparison of solutions is made based on two criteria; qualitative comparison of LP and ES solutions and computational efficiency of these approaches. In the following, all voltages are in p.u., reactive compensation is in MVAR and tap positions are given in terms of the secondary to primary turns ratios.

3.5.3 Quality of Solutions - Comparison

Referring to the results in Tables 3-4 and 3-5, there are 3 distinct qualitative improvements in the solutions proposed by the expert system. First, the final voltage profile is higher. This follows because the expert system is attempting to return the voltage violating buses to a 1.0 p.u. level in Cases 1-3 and to a 0.98 level in Cases 4-5. The LP, on the other hand, is adjusting controls to relieve constraint violations, implicitly assuming that all solutions within the feasible region are qualitatively equivalent. Second, the LP may suggest small infeasible adjustments to controls (e.g. RC 19 0.4% adjustment in Case 4). The expert system avoids this by considering one control at a time. Third, discrete constraints are easily incorporated into the expert system and it is possible to suggest practical control settings. Integer LP algorithms for considering discrete adjustments suffer from computational inefficiency.

The LP is superior at minimizing the number of controls used for the more severe voltage problems. This is expected as the LP algorithm is optimized for this purpose. There may be situations where this is an overriding concern. Notice, that as the problems become more severe the quality of the expert system solution decreases in that more controls are used then is probably necessary. This implies it is natural to expect assistance from numerical algorithms as severe voltage violations occur.

Table 3-4 Suggested Control Actions

Case	Control Actions		
	ES	LP	ES/LP
1	RC29=0.090	RC29=0.009	--
2	RC29=0.070	RC29=0.047	--
3	RC24=0.100 RC29=0.100 Tap26=1.040 Gen8=1.050	RC29=0.069 Tap26=1.031	--
4	All RCs =0.100 Tap16=1.040 Tap26=1.040	RC19=0.0004 RC29=0.056 Tap16=1.006 Tap26=1.038	All RCs =0.100 Tap16=1.038 Tap26=1.066 Gen13=1.100
5	All RCs =0.100 Tap16=1.040 Tap26=1.040 Gen8=1.040 Gen13=1.100	RC19=0.071 RC24=0.022 RC29=0.069 Tap16=1.003 Tap26=1.028	All RCs =0.100 Tap16=1.028 Tap26=1.0003

Table 3-5 Final Voltage Profiles

Case	Method	16	17	18	19	20	21	22	23	24	25	26	29	30
1	ES													.986
	LP													.949
2	ES												.988	.959
2	LP												.975	.949
3	ES									.990	.990	.993	1.010	.981
3	LP									.957	.958	.952	.977	.949
4	ES	.997			.988	.985				.988	.983	.982	1.001	.976
4	LP	.950			.950	.951				.954	.955	.952	.972	.949
4	ES/LP	1.029			.993	.990				.993	.986	.983	1.003	.978
5	ES	1.011	.984	.990	.987	.983	.989	.990	.995	.991	.995	1.00	1.020	.993
5	LP	.950	.954	.959	.952	.950	.956	.956	.961	.950	.956	.951	.973	.949
5	ES/LP	.963	.971	.976	.973	.970	.976	.977	.980	.977	.979	.975	1.002	.975

This combination of ES and LP methodologies allows a compromise between the heuristic and number-crunching approaches. The type of assistance the expert system should provide the LP remains an area for investigation.

3.5.4 Computational Efficiency - Comparison

In order to evaluate the computational efficiency, CPU times were measured for all approaches. In the following, all measurements are either in seconds(s) or milliseconds(ms).

It is clear from Tables 3-6 and 3-7 that the expert system is an efficient approach for the examples presented. One advantage results from iterations (with the corresponding loadflows) the LP must perform in order to converge to an actual solution for the non-linear power

system. This problem will worsen as the dimension of the problem grows and both the LP and the loadflow slow down. The expert system processing time is dependent only on the data that enters working memory. Thus, it can be made relatively independent of system size (assuming sensitivities have been determined off-line). The system is quite dependent on the number of voltage violations as predicted in section 3. This dependency is further analyzed in the following section.

3.5.5 Analysis of VCES Processing Costs

For the purpose of analysis, the processing costs of VCES are broken down to four basic tasks: Rule Firing overhead(RF), Conflict Resolution(CR), Changes to Working Memory overhead(WMC), and Pattern Matching(PM). It is reminded that PM is mainly to perform the intercondition tests. Table 3-8 summarizes the processing costs of the selected scenarios. Note for each case the number of operations of each category is given first, the processing time(msec) and the percentage relative to the total CPU time then follow. For example, Case 1 requires 105 RF operations, which takes approximately 40 msec or 5% of the total CPU time. The following observations are made based on results of Table 3-8.

Table 3-6 - CPU Measurements

Case	ES	LP	# of Iterations	LF verification & sensitivity calculation
1	0.78s	2.96s	1	15.33s
2	1.21s	5.88s	2	27.81s
3	2.93s	5.99s	2	28.11s
4	6.02s	8.98s	3	46.28s
5	9.00s	9.03s	3	38.32s

Table 3-7 - CPU Measurements- LP/Expert System Solution

Case	ES	LP	# of Iterations	LF verification & sensitivity calculation
4	5.71s	3.00s	2	31.01s
5	10.19s	6.08s	3	43.33s

Table 3-8 - Breakdown of Processing Costs for VCES

Case	RF			CR			WMC			PM			Total
1	105	40	5	4610	599	78	309	133	17	NA	NA	--	780
2	109	42	3	4731	615	51	319	137	11	NA	416	34	1.21s
3	212	82	3	5579	725	25	519	223	8	NA	1900	65	2.93s
Expert System and no LP													
4	245	94	2	6571	854	14	699	301	5	NA	4771	79	6.02s
Expert System and LP													
4	312	120	2	11703	1521	27	811	349	6	NA	3720	65	5.71s
Expert System and no LP													
5	126	126	1	8762	1139	13	925	398	4	NA	7337	82	9.00s
Expert System and LP													
5	150	181	2	18954	2464	30	1179	507	6	NA	5038	62	8.19s

1) In all cases the rule-firing(RF) and changes to working memory(WMC) overheads represent a small portion of the total processing time.

2) In the first two cases, the conflict resolution time dominates. The number of low voltages is small for cases 1 and 2, and therefore few intercondition tests need to be performed, as can be seen from the discrimination net. The large number of CR count has to do with data refreshing after a new set of data is obtained from the database. In the 30-bus case, the present implementation leads to many instantiations of the same rule which changes working memory elements.

3) In the last three cases, the pattern matching time increases very significantly. This is expected since the number of low voltages increases and many intercondition tests are necessary to search through the reactive

resources and estimate the right amount of control adjustments.

4) In Cases 4 and 5, the number of conflict resolution operations almost doubled when the linear program is applied. Note, that the LP solved the voltage problems but it was called upon twice. As a consequence, the number of rules that entered the conflict resolution step was approximately doubled.

In general, as the number of problem buses grows, say in a large power system, more rules will be included in the CR set. In this case, the time for conflict resolution, i.e., finding the rule ranked highest in a specified priority system, increases linearly. As for the pattern matching time, it grows exponentially as the low voltage profile spreads out.

3.5.6 Remarks

This chapter presents an analysis of an expert system for voltage control. The ES approach was compared to a numerical approach in order to validate the capability of the heuristic method. Computational issues of rule-based programs are analyzed in detail. The forward chaining mechanism is divided into basic tasks whose computational costs are measurable. Based on the breakdown, it is possible to gain insight into the design of efficient expert systems.

To focus on the voltage problem, the following design rules are useful.

1) If the voltage problem is confined to an area, a neighborhood of the area with reactive resources can be identified for the solution. It is unnecessary to consider the whole system. Again, this is intended to reduce the pattern matching time.

2) Avoid very inviting conditions, which tend to increase the number of intercondition tests. The voltage problem can be grouped suitably to avoid a large amount of matching between problem buses and reactive controllers.

3) In some situations, it is not necessary to resolve the conflict among rules. Note that CR step limits the number of rules that can be fired to one, which does not serve any purpose if several actions can be taken at the same time.

For on-line operation purposes, an ES would have to meet the efficiency requirement. This section has provided some guidelines for the improvement of expert system implementation.

3.6 Further Work

The proper role of heuristics and number-crunching algorithms has not been completely established. It is clear

that for many situations a heuristic approach is preferable. Also, heuristics may not solve certain complex problems well and then reliance on algorithmic approaches will be necessary. Between these two extremes, there are borderline cases where the exact tradeoffs and compromises need to be investigated.

Voltage problems occur relatively rarely for most utilities. This creates a situation where there may not be sufficient expertise to generate useful heuristics. In this case, automatic knowledge acquisition becomes an important area for investigation. Heuristics can be generated from either simulations or the vast amounts of raw data available from utilities. Automating performance evaluation and the generation of heuristics as the first steps in automatic knowledge acquisition is on-going research at the University of Washington.

4. COMPUTATIONAL ASPECTS OF RULE-BASED SYSTEMS

Current work on expert systems has focused primarily on knowledge representation issues and specific applications. Up to the present, little has been reported on the analytical aspects of the computational efficiency of expert systems as real-time operational tools. Is a given rule-based program efficient enough to solve a specific class of problems? This question has to do with the implementation. Notable implementation issues include rule structure, data structure, software language, and hardware support. The initial effort in efficient processing of rule-based systems was the RETE algorithm which utilizes a faster pattern matching method [30]. Improvements on the RETE algorithm have been proposed [31]. Other work has continued in the area of parallel processing of rule-based systems [32-35]. Many expert system shells contain efficient algorithms for implementing production systems [4]. These studies all concentrate on improving the efficiency of expert system development tools. Literature on analysis and design guidelines for efficiency in an engineering sense is scarce. Reference [28] provides some general guidelines for efficient OPS5 implementation of rule-based systems, however, this work doesn't evaluate computation but only suggests approaches for alleviating bottlenecks once they

are found. The lack of literature is due partly to the difficulty of analytical representation of the expert system's knowledge base. Furthermore, expert systems tend to be large, complex and non-procedural and, therefore, a detailed analysis is often deemed impractical.

This chapter is concerned with systematically evaluating computational performance of a developed expert system and generating criteria for performance improvement. The applications that are of principal concern here are ES which deal with meeting operational constraints in an on-line environment, e.g., maintaining service to customers, alleviating voltage violations, etc. Furthermore, we are interested in forward-chaining inference engines, similar to OPS5 (pattern match, conflict resolution, act), although the approach is, in general, applicable to other rule-based systems. It is important to know what is the slowest response time to a disturbance ('emergency' condition) that can occur for on-line software. That is, if the worst-case scenario is fast enough then real-time performance can be guaranteed. Thus, an upper bound on the time required to solve any single scenario of the problem domain is derived. (In the examples presented, a single scenario corresponds to one faulted line section or one problem voltage scenario). An algorithm for determining the upper bound using the extant expert system and appropriate description of the application domain is proposed. The conditions required for

this upper bound to be hard (achievable) are identified and numerical examples presented to show the closeness of this bound to the actual worst case. Issues involved with improving performance are discussed. Improvements are implemented and numerical examples are used to highlight the results. Suggestions for further work are discussed.

4.1 A Computational Model

In this section, a model for determining computational costs is proposed. The purpose of this model is to provide an analytical framework for discussing efficiency issues. Section 3.4.1 described the execution of a forward-chaining ES. Briefly summarized, there are three major steps which are performed for every rule firing: pattern matching - find all rules which are satisfied, conflict resolution - determine which rule to fire among those rules which are satisfied, and act - execute the conclusion of the selected rule which may include changes to WM or other processing functions. Fig. 4-1 illustrates the iterative process of rule firings. The computational costs of each of the blocks in Fig. 4-1 will be discussed.

For pattern matching, define the following quantities:

w_i ≡ Working Memory Element (WME) i

$W_m = \{w_i\}$ $\quad i \in 1,2\cdots m, \quad$ i.e. $W_m = \cup_i w_i$

$N_j = \{n_i\}, \quad n_i \in \underset{d}{\mathbf{X}} W_m$

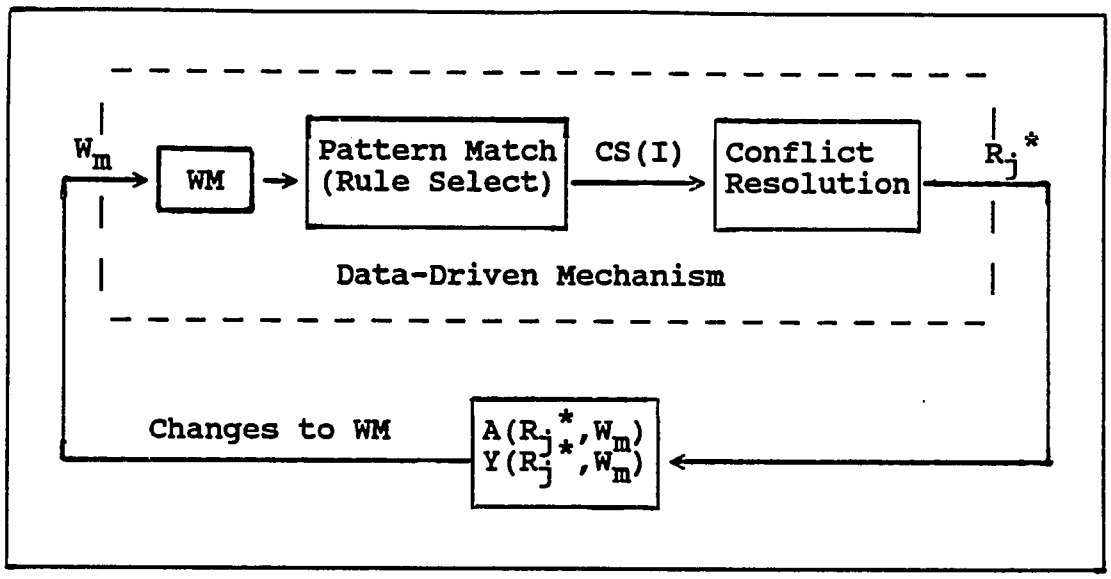


Fig. 4-1 Iterative Process in Data-Driven System

where $\mathbf{X}_d$ represents the cross-product over W_m of dimension d. For example in Fig. 4-2, if d=2 then $n_i \in W_m \text{ X } W_m$, where m indexes the succession of working memory changes. N_j represents node j in the discrimination net and n_i defines the WME pairs which have been matched at that node. At a terminal node in the network, each $n_i \in N_j$ represents a rule instantiation R_j. Let N(I) be the union of all N_j at iteration I so that N(I) defines the current state of pattern matching for the discrimination network. Now, the action part of a rule instantiation R_j may be associated with a set of working memory changes defined by $A(R_j, W_m)$. In addition, there may be other functions performed in the RHS of the rule, e.g., performing a loadflow in power system applications. These functions are represented as $Y(R_j, W_m)$.

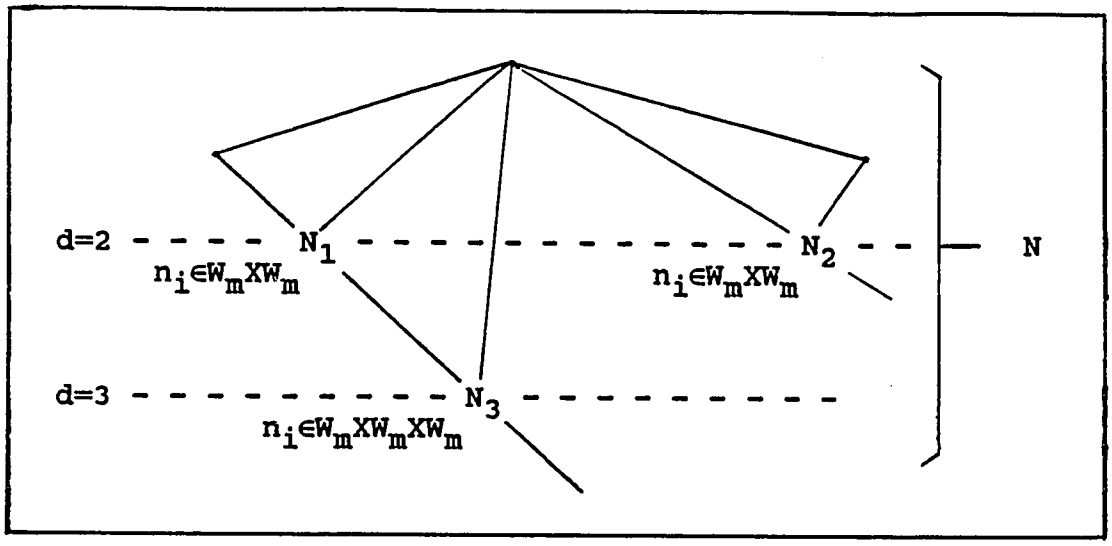


Fig. 4-2 State Space of Discrimination Net

The process of reaching a decision requires chaining together rules to form a line of reasoning. It is possible to define relationships between rules based on their conditions and actions so that, a search tree, for example, would be constructed to indicate the possible lines of reasoning. However, the size of this tree would be huge for all but the simplest of examples. Rather than finding all these lines of reasoning and enumerating costs along each path, an unordered set of rule firings will be defined. Let's define R as this set of rule firings, i.e., $R = \cup R_j^*$ and C as the total cost of computation, so that:

$$C = \sum_{R_j^* \in R} C_1(R_j^*) \qquad (4.1)$$

where $C_1(R_j^*)$ is the cost of firing rule instantiation R_j and the asterisk denotes R_j as the selected rule of conflict

resolution (see Fig. 4-3). Note that the cost of firing R_i^* and R_j^* may be different. Let the cost of conflict resolution be given by $C_2(N(I))$. Also, define $C_4(w_i;N)$ as the cost of matching w_i for a given discrimination network N and $C_{5k}(w_i)$, which is assumed to be independent of the current state of WM, as the cost of testing attribute k on w_i. K_i is the set of attribute tests for the WME class corresponding to WM element i. So, the cost of creating w_i is:

$$C_3(w_i;N) = \sum_{N_k \in N(I)} C_4(w_i;N_k) + \sum_{k \in K_i} C_{5k}(w_i) + \text{Overhead} \qquad (4.2)$$

where $C_4(w_i;N_k) = \sum_{w_j \in N_k} c_4(w_i,w_j)$, c_4 is the cost of matching working memory elements w_i and w_j at node k and $C_4(w_i;N_k) = 0$ if w_i does not encounter node N_k. Furthermore, allow $C_6(Y(R_j,W_m))$ to be the cost of executing the actions of R_j, $Y(R_j,W_m)$. Then $C_1(R_j^*)$, the cost of firing rule instantiation R_j, is:

$$C_1(R_j^*) = C_2(N(I)) + \sum_{w_i \in A(R_j,W_m)} C_3(w_i;N) + C_6(Y(R_j,W_m)) + \text{Overhead} \qquad (4.3)$$

4.2 Bounding Computational Costs

In placing an upper bound on computational time, the time used will be separated into worst case time for pattern matching and conflict resolution, and the line of reasoning which causes worst case costs to be chained together.

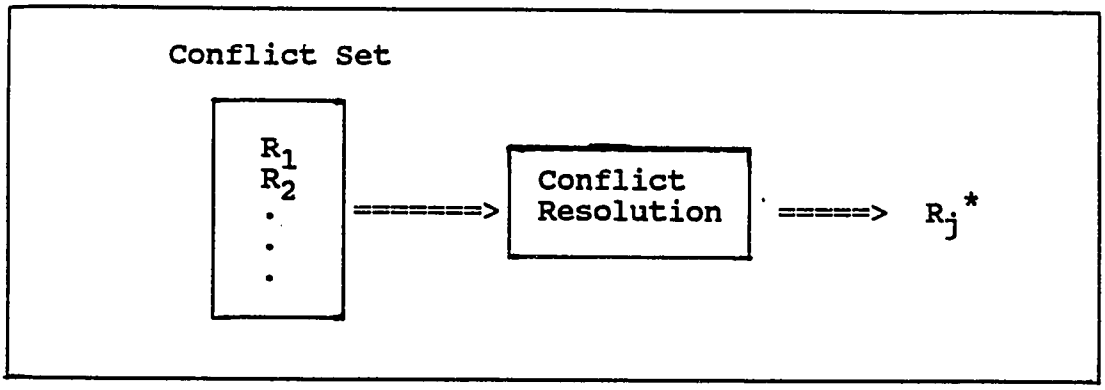


Fig. 4-3 Conflict Resolution Map

For pattern matching, the slow down is primarily when large numbers of intercondition tests, corresponding to relating several different elements in WM, must be performed, i.e., when $\|N_k\|$ is large where $\|\cdot\|$ denotes the cardinality of a set. In order to find the computation time, the following must be known about WM and the rule structure: the number of WMEs which satisfy rule conditions at node k, $\|N_k\|$, and the subnetwork which defines the relationships between the various WME classes, i.e., the path followed in the net based on the rule conditions. It will be seen later that these relationships are implicitly described in the expert system and thus do not need to be explicitly represented in order for an algorithm to find an upper bound on pattern matching.

The cost of conflict resolution is dependent on the number of rules in the conflict set and the scheme used for rule selection. For VCES and CRAFT, the following priority

was used: (1) Rule with the most recent 'goal' element (the element which satisfies the first condition), (2) Most specific rule, (3) Rule with the highest rank (either a constant or variable value based on WMES and global variables assigned to each rule by the programmer), (4) Rule which has been in the conflict set the longest. Since, practically the cost of conflict resolution is relatively independent of whether only criterion (1) applies or all four criteria from above must be applied, the worst case will be assumed to be only a function of the size of the conflict set. That is, the worst case occurs when the conflict set is largest.

Finding the worst case line of reasoning is more difficult. For instance, the scenario that causes the longest chain of reasoning (most rule firings) does not necessarily result in the slowest response by the expert system. Also, with the exception of very simple applications, it is impractical to exhaustively search all lines of reasoning in order to determine the worst case. On the other hand, if one determines the number of times a rule may fire and which rules exclude the firing of other rules, then each possible chain of reasoning does not need to be investigated, instead the worst case contributions from individual rules can be combined without concern for the ordering of rule firings. The proposed algorithm will determine these quantities and relationships by subdividing

the rule-base and explicitly defining relationships between working memory elements. These ideas are presented formally in the next section.

4.2.1 The Worst Case Computational Cost

The condition part of a rule can contain a number of attributes, which are organized into a net (Fig. 3-5). In the worst case, all the attributes of a WME w_i will be tested during pattern matching. This corresponds to the case when w_i is in all possible nodes at level d=1 in N (Fig. 4-2). Let this be represented as:

$$\bar{C}_5(w_i) \geq \sum_{k \in K_i} C_{5k}(w_i) \tag{4.4}$$

The number of intercondition tests at a node N_k equals the product of the number of WMEs at each incoming node, so to determine the tests performed for a WME w_i we need to know the number of elements at each node as well as the nodes encountered by w_i. Define the bounds:

$\bar{N}_k$ = Node state when the # of elements, $\|N_k\|$, is at a maximum (maximized over all possible scenarios and N_k is encountered by w_i).

And for the entire network:

$$\bar{N}(w_i) = \bigcup_k \bar{N}_k$$

Now let $\bar{C}_4$ bound the cost of intercondition testing:

$$\bar{C}_4(w_i) = \sum_{N_k \in \bar{N}(w_i)} C_4(w_i; N_k) \tag{4.5}$$

Then if we define $\bar{C}_3$ as:

$$\bar{C}_3(w_i) = \bar{C}_4(w_i) + \bar{C}_5(w_i) + \text{Overhead} \qquad (4.6)$$

It follows that:

$$C_3(w_i;N) \leq \bar{C}_3(w_i)$$

Now observe that a rule node is just a special case of other nodes in the network so that the maximum number of rule instantiations and thus the maximum size of the conflict set occurs under $\bar{N}$. Then the cost of conflict resolution is bounded, $C_2(N(I)) \leq \bar{C}_2(\bar{N})$. Furthermore, allow $\bar{C}_6(Y(R_j))$ to be the maximum cost over all states of WM and I/O of executing $Y(R_j,W_m)$. Combining these with equations 4.4 - 4.6 we arrive at:

$$\bar{C}_1(R_j) = \bar{C}_2(\bar{N}) + \sum_{w_i \in A(R_j)} \bar{C}_3(w_i) + \bar{C}_6(Y(R_j)) + \text{Overhead} \qquad (4.7)$$

$$(C_1(R_j;N) \leq \bar{C}_1(R_j))$$

Recall that R is the set of all rule firings for a given scenario. Define $\bar{R}$ as the worst case set of rule firings in the following way. If $R_i \in R$ for any scenario then $R_i \in \bar{R}$ unless R_i, R_j are exclusive, i.e. $R_i \in R \Rightarrow R_j \notin R$. In that case, if $\bar{C}(R_i) \leq \bar{C}(R_j)$ only $R_j \in \bar{R}$. (If necessary exclusiveness is defined over sets of rules instead of the individual rules R_i and R_j above.) Finally, the computational cost C is bounded by:

$$C \leq \sum_{R_j \in \bar{R}} \bar{C}_1(R_j) \qquad (4.8)$$

4.2.2 Computational Bound: Algorithm

It was seen that by examining the possible pattern matching, the conflict set and the characteristics of rule firings, an upper bound could be placed on the computation time. This approach required determination of several quantities: the number of possible attribute tests for a given WME, the limits on the numbers of WMEs which satisfy various conditions in a rule, the subnetwork defined by the rule conditions which apply to w_i, the limits on the cost of conflict resolution, and the relationships between rules which apply to different operating states. In this section, an algorithm is described which determines these quantities.

The worst case pattern matching occurs when the most possible elements are satisfied at each node in the network. By creating WMEs so that the maximum number of elements satisfies each attribute test and each intercondition test for a rule, i.e. the maximum number of elements for all N_k, a discrimination-net which always achieves or exceeds the most possible pattern matching is realized. To determine the number of elements which satisfy any individual conditions within a rule one must look at the characteristics of the problem domain. For instance, in VCES if a rule condition matches all PQ buses then it is known that there are 24 PQ buses in the IEEE 30-bus system. More involved is the computation of the number of WME combinations which satisfy multiple rule conditions, i.e. deeper nodes in the discrimination network, since

interrelationships between WME classes must be determined. However, many intercondition tests are searching for only one match. For example, in CRAFT if a rule needs to know the status of the pcb at the strong end of a specified line there is only one WME combination describing the information needed, i.e. the pcb information and the specific transmission line under consideration. Other intercondition tests may be satisfied by all combinations of WMEs. For example, in VCES several rules contain intercondition tests which match each bus that has a voltage violation with available reactive controllers. This observation simplifies greatly computing the maximum value of WME combinations at deeper nodes in the network. Practically, the WM state that creates the worst case for the creation of a particular WME, w_i, will not necessarily create the worst case situation for another WME, w_j. In other words, several different WME states may need to be created to find the worst case bound. Furthermore, one has to model the physical system condition (input data) so that the maximum number of working memory elements are created.

The above procedure establishes $\bar{C}_4$ and $\bar{C}_5$. The worst case cost of pattern matching is obtained by combining all worst case costs of working memory changes in $A(R_j,W_m)$. Also notice that, the conflict set is the largest possible since the maximum number of instantiations occurs for each rule. (This follows since a satisfied rule node is just a

particular type of node in the network). Furthermore, it is assumed $C_6(Y(R_j,W_m))$ is known. So that now, the worst-case cost of firing any rule instantiation is known.

In order to complete the algorithm $\bar{R}$ must be determined. This process can be aided by the structure of the rule-base which is typically broken down into groups or tasks. These tasks form a natural partitioning of the rule-base and the worst case for this smaller rule-base is determined. For each individual task, we can allow the worst case for all of the rules within that task. The number of rule firings is found by the rule instantiations (determined above by allowing the maximum number of WMEs to be created) and examining which rules within a task exclude other rules from firing. Apparently, knowledge of the application domain is critical to know which rules are exclusive; for instance, a particular bus can only have either a high or a low voltage problem not both. So only the high voltage rules or only the low voltage rules will be included in the worst case set of rule firings. This determines the maximum cost associated with completing a task. Now, the number of executions of the task is determined. There will be a rule or possible a set of rules which can create this task. The maximum number of instantiations of a rule defines the possible number of times this rule can create the task. However, this must consider the exclusive rule set to obtain an accurate upper bound on the task executions. Alternatively, system

constraints may provide a simpler approach. Consider the control effect task in VCES, there are 14 controls in the IEEE 30-bus and thus the task can only be executed 14 times, once for each controller.

It is assumed the application domain is understood well enough so that all these relationships are known. The various components of the bound (Eq. 4.8) have now been determined. In the following, the algorithm for determining the worst case bound is summarized.

The Algorithm

1. Select a task. This is most easily done if the rule-base has been written in modular manner so that any module can be selected. Otherwise, rules should be grouped based on the amount of mutual interaction in order to simplify rule relationships.

2. Modify WM so that the maximum number of rule instantiations for each rule within the chosen task can be found from the conflict set.

3. Select a rule instantiation within the chosen task. The same procedure used to obtain the state of WM in (2) from above can be used to find the worst case pattern matching, $\bar{C}_4$, for each WME change within $A(R_j,W_m)$. Conflict resolution, C_2, is a maximum based on the state of WM from above. Also find the maximum cost of $Y(R_j,W_m)$. (Recall $C_6(Y(R_j))$ is needed to calculate the bound.)

4. Now compute $\bar{C}_1$ for each rule instantiation.

5. Repeat steps 2-4 for each rule within the selected task and for each possible execution of this task.

6. Add the computation time of a rule instantiation to the worst case cost only if it is more expensive than all rules with which the rule is exclusive, i.e. form $\bar{R}$.

7. Repeat steps 1-6 for each task.

4.2.3 How Good is the Bound?

There are several factors which will cause the derived bound to exceed the actual worst case. These are:

(1) The computed worst case may not be realizable since some constraints imposed by the physical system being operated are not modeled within the expert system. For example in the simulator portion of CRAFT, the worst case scenario may be when all automatic switching operations occur at once on a particular line. However, planning engineers in order to avoid racing between switch operations will stagger automatic switch timings.

(2) Attributes are assumed to be independent. $\|N_k\|$ was determined by allowing maximum intercondition testing to take place for the creation of any WMEs. However, there may be attributes which are interdependent and the algorithmic determination of $\|N_k\|$ will be somewhat higher than the true value.

(3) When determining which rules exclude other rules only other rules within a given task are considered. This, in essence, assumes worst case scenarios in different tasks will occur together. This is not true in general, so that the worst case bound may be more expensive than possible.
(4) Conflict resolution is assumed to be independent of the conflict resolution scheme. This will not always be true. Conflict resolution computations, however, are normally fewer than pattern matching and this difference should not be large.

When the above situations are not encountered it is possible for the processing of a scenario to reach $\overline{C}$.

4.3 Improving Computational Performance

In [28], several suggestions for improving efficiency are given. Briefly these are:

(a) Avoid conditions that match many WMEs
(b) Avoid big cross-products between conditions
(c) Avoid frequent changes to matched conditions
(d) Make matching individual condition elements faster by ordering the most restrictive attribute tests first
(e) Limit the size of the conflict set
(f) Call user-defined algorithmic procedures.

The problem with these guidelines is that one, some of them are contradictory, e.g., avoiding frequent changes to matched conditions may cause big cross-products, two, they don't indicate where the principle computational problem lies, and three, they don't provide a criterion for knowing when changes have actually improved computational performance. A different approach to improving computational efficiency is proposed here. The question of improved performance has been reduced down to more fundamental quantities; the computations taking place are explicitly represented so the problem and possible solutions are better identified. Furthermore, worst case computational cost is proposed as a criterion which can be used as a performance guide.

In essence, there are four interrelated quantities which comprise the computations which are most affected by implementation: R the set of rule firings, C_2 the conflict resolution operations, C_4 the cost of intercondition testing, and C_5 the cost of attribute testing. To begin, $\|R\|$ is reduced by calling user defined functions, (e) above, and possibly by avoiding frequent changes to WMEs by performing many changes or operations within a single rule, (c) above. However, calling user defined functions may just shift computation from the inference engine to $Y(R_j,W_m)$, so the tradeoff must be investigated. Properly defined data structures can decrease excessive rule firings. For example,

in CRAFT there is a rule condition which matches the closest device. By defining an ordering on the devices, this can be tested for with a single rule condition. An alternative representation using a standard tree data structure but linked through WMEs would require repeated rule firings to search for the closest device. This search for a closest device is repeated at several steps in the decision-making so the increased cost would be significant. Otherwise $\|R\|$ is dictated by the problem domain and the associated heuristics.

C_4 for a given rule can be reduced by varying the ordering of rule conditions, (b), or using more selective conditions, (a). Usually, a more effective approach for reducing pattern matching operations is to define data structures that avoid searching through lists of WMEs to find a match. For example, one can combine two WME classes, say sensor and device in CRAFT, so that the information contained within these WMEs, in this case the sensor associated with a particular device, is available as one entity without searching.

Criterion (d) suggests ordering attributes to reduce C_5. In a worst case scenario, this should not be a significant change. Few programming styles will result in excessive attribute testing. Finally, criterion (e) simply suggests limiting $\|CS\|$, thus decreasing C_2. Typically, a large conflict set is more a result of incorrectly using conflict

resolution then a situation where several heuristics within the problem domain are applicable. In certain cases (e.g., the updating working memory example presented in Chapter 3), no conflict resolution is necessary. An improvement in conflict resolution can be obtained by simplifying (essentially turning off) conflict resolution during the rule firings where conflict resolution is unnecessary. At this time, the savings of turning off conflict resolution at appropriate points in the reasoning have not been investigated in either CRAFT or VCES.

The algorithm for determining the worst case bound reveals which working memory elements for pattern matching or which rules in a chain of reasoning make large contributions to the computation time. To analyze proposed changes in implementation, the change can be incorporated into the rule structure and the new worst case found in order to determine if an improvement has been obtained. It is a topic of future research to see if the algorithm can be augmented to recommend changes in rule structures.

4.4 Numerical Examples

In this section, the computational efficiency of VCES is analyzed. A sample power system commonly referred to as the IEEE 30-bus power system is used. To begin, a task composed of 9 rules for determining the effect on neighboring buses of a controller was utilized to illustrate the algorithm and

possible improvements. This task was chosen since it is representative of the increased computation associated with more widespread voltage problems. Table 4-1 shows the upper bound for this task. (All time measurements are indicated by s and ms for seconds and milliseconds, respectively). Notice, that the number of executions is limited by the number of controls (14). So, that one expects the worst case to worsen with an increasing number of controls. Table 4-2 shows the breakdown of costs in terms of pattern matching, conflict resolution and the actions taken in rule firings including overhead for the worst case single execution of control effect. It can be seen that the dominant cost (88%) for this task is pattern matching. Using this as a guide, a number of different rule implementations were developed by varying the data structure and the ordering of conditions in the rules. A description of these rule changes is given in Fig. 4-4.

Table 4-1 Worst Case Bound for Control Effect Task in VCES

Scenario	CPU Time
One Execution	2.54s
Multiple Executions (14)	14.20s

Table 4-2 Breakdown of Computational Costs for Worst Case of Control Effect Task

Operation	# of Executions	Time	% of Total
Actions $Y(R_j, W_m)$	49	18.8ms	1
Conflict Resolution C_2	2101	273.1ms	11
Pattern Matching C_4	194	2.25s	88

```
Original:
1 (task id=control_effect);
2 (voltage_violator)
3 (reactive_controller name=1.subject; )
4 (bus name=3.location; )
5 (bus type=pq; name=2.violator; status<>checked; )
6 (jacobian row=5.name;)
7 (compensator row=5.name; id[1]=1.subject; )
Structure 1: Original data structure with reordering of
             conditions.
1 (task id=control_effect);
5 (bus type=pq; status<>checked; );
2 (voltage_violator violator=5.name; );
4 (bus )
3 (reactive_controller location=4.name; name=1.subject; )
6 (jacobian )
7 (compensator row=6.name; id[1]=1.subject; )
Structure 2: Original data structure with reordering of
             conditions.
1 (task id=control_effect);
3 (reactive_controller name=1.subject; )
6 (jacobian )
7 (compensator id[1]=1.subject; row=6.name; )
4 (bus name=3.location; );
2 (voltage_violator violator=7.row; );
5 (bus type=pq; name=2.violator; status<>checked; );
Structure 3: Single WME class for conditions: 2 6 7
1 (task id=control_effect; );
2 (voltage_violator id[1]=1.subject; );
3 (reactive_controller name=1.subject; )
4 (bus name=3.location; )
5 (bus type=pq; name=2.violator; status<>checked; )
```

Fig. 4-4 Representative Rule Structures in Control Effect Task

For each of the rule structures indicated in Fig. 4-4, the upper bound on computations was determined. These bounds were compared to the computation time spent in a typical example (two voltage violations). Finally, measurements were made on the entire rule-base of VCES using the original rule structure and a modified rule structure incorporating the ideas of reducing pattern matching by combining WME classes and reordering conditions. From the results of Table 4-3, it can be seen that an inappropriate ordering can create excessive matching leading to very slow response (structure 1 versus structure 2). Also, Table 4-3 indicates that a carefully designed data structure can decrease matching significantly (structure 3). Worst case response improved roughly 25% by making a few modifications to the rule-base in VCES (Table 4-4). In attempt to find the actual worst case for Table 4-4, the computation time of several potentially slow examples was measured and the worst among these was used as the 'worst case found'. Note in Tables 4-3 and 4-4 that the average case was improved when the worst case cost decreased. So the proposed method gives some insight into rule structure and how to modify rules.

Table 4-3 Measurements on Rule Structures for Control Effect Task

Rule Structure	Upper Bound	Typical
Original	2.54s	0.08s
Structure 1	36.25s	0.35s
Structure 2	1.42s	0.06s
Structure 3	1.13s	0.05s

Table 4-4 Worst Case Scenario for VCES

Rule Structure	Worst Case Found	Upper Bound	Typical (Averaged)
Original	55.8s	67.4s	4.09s
Modified Rule Base	43.1s	54.8s	3.50s

4.5 Remarks

The numerical examples indicate that very similar rule-base implementations can have significantly different computational characteristics. This suggests that careful analysis must be applied to any rule-based system, such as CRAFT, for which fast response is important. To begin, the ordering of rule conditions can greatly change the number of intercondition tests that must be performed. Tradeoffs between frequently changing WME classes and inviting conditions should be balanced. For example in VCES, many rule conditions will be satisfied by several WMEs but

information about a voltage violator and a load bus is changing fairly often so that the rule conditions testing these WME classes should come later. Another powerful method for improving efficiency is to combine WME classes so that much of the intercondition testing is avoided. This can only be done to the extent that the WME classes are closely related.

A few weaknesses in the proposed approach are apparent. Given an expert system, it is time-consuming to determine the upper bounds and to improve the design to achieve a satisfactory implementation. Clearly, if this process could be automated the approach would be more useful. Furthermore, the worst case may represent a pathological case and thus not the best measure of speed. However, the worst case provides useful information as to where the bottleneck for the expert system processing is, which should help expert system developers improve the quality of implementation.

5. CONCLUSION

This dissertation has reported on the development of two expert systems: one, CRAFT, for locating line faults and restoring customers, the other, VCES, for taking corrective actions in order to alleviate voltage violations. The quality of knowledge contained within the expert system must be validated thoroughly. In CRAFT, the validation process was performed by system dispatchers. For VCES, simulation of the interconnected power system checked against operating constraints was used to validate performance. The two systems represent different philosophies of expert system applications. CRAFT is the traditional expert system application recording the experience of engineers in the field. VCES utilizes the rule-based programming technique to implement ideas that were obtained partly from dispatcher experience, partly from mathematical analysis and partly from the experience of extensive simulations. Extensive testing of VCES indicates this is a viable approach. Furthermore, there are many problem domains where this type of approach is necessary, for instance, those problems which occur rarely so that individual experts seldom develop yet empirical approaches still apply. It is felt that CRAFT and VCES establish the feasibility of rule-based approaches for on-line power system operational aids.

There are areas of improvement for both CRAFT and VCES. Neither ES has actually been implemented in a control center and the experience obtained from actual operation and implementation will undoubtedly indicate some changes. CRAFT is limited by several assumptions. The most limiting of these is reliable data, particularly for unsupervised data points (pseudo-points) where changes in data are input manually. Work is on-going at the University of Washington in order to loosen this assumption of reliable data. Another restriction of CRAFT is that it limits scenarios to within a single line. More widespread problems, e.g., resulting from miscoordination of circuit breakers, has not been considered but may certainly arise in actual operation. VCES is seen to be effective for a variety of voltage problems, however, the more severe problem of voltage collapse is not considered. Furthermore, severe voltage problems where reactive power sources are barely sufficient or insufficient (and require load shedding) to alleviate the voltage violations may not be solved by the heuristic approach.

Many important ES development issues exist including choice of problem domain, data management, interfaces to other problem solving software (e.g., loadflow), verification of problem heuristics and efficient implementation. To begin, performance criteria need to be developed in order for expert systems to become useful power system tools. In on-going work, the automation of

qualitative performance analysis is being considered. In this thesis, computational performance has been scrutinized. An upper bound on computations was developed which gave an important indicator for real-time performance. The indicator was used to reduce computations of rule-based implementations. Many other considerations remain to be addressed. Among these, several areas appear promising as directions for future research, they are:

- Incorporation of domain knowledge into databases (i.e. expert databases).
- Automating knowledge acquisition. Including: automation of performance evaluation (developing expert critics), generation of new heuristics from extensive simulations or through existing data.
- Augmenting the worst case bound algorithm to suggest implementation changes which will improve the computational performance of the developed expert system.
- Investigating the proper method of combining heuristic and numerical approaches.
- Development (or investigation) of ES tools particularly suited for power system applications.

6. REFERENCES

[1] E. A. Feigenbaum, "Knowledge Engineering for the 1980's," Computer Science Department, Stanford University, 1982.

[2] R. Davis, B. Buchanan and E. Shortliffe, "Production Rules as Representation for a Knowledge-Based Consultation Program," Artificial Intelligence, Vol. 8, 1977, pp. 15-45.

[3] J. McDermott, "R1: A Rule Based Configurer of Computer Systems," Artificial Intelligence, Vol. 19, 1982, pp. 39-88.

[4] F. Hayes-Roth, D.A. Waterman and D.B. Lenat, Building Expert Systems, Addison-Wesley Co., 1983.

[5] J.H. Greismer, et. al., "YES/MVS: A Continuous Real Time Expert System," Proceedings AAAI-84, pp. 130-136.

[6] S. D. Campbell and S. H. Olson, "WX1 - An Expert System for Weather Radar Interpretation", Coupling Symbolic and Numerical Computing in Expert Systems, 1986, pp. 329-348.

[7] B. F. Wollenberg, "Feasibility Study for an Energy Management System Intelligent Alarm Processor", IEEE Transactions on Power Systems, Vol. PWRS-1, No. 2, May 1986, pp. 241-247.

[8] S. N. Talukdar and E. Cardozo, Artificial Intelligence Technologies For Power System Operations-- Some Demonstrations And An Assessment of Opportunities, Final Report, RP 1999-7, EPRI, Aug. 1985.

[9] K. Komai, T. Sakaguchi and S. Takeda, "Power System Fault Diagnosis with an Expert System Enhanced by the General Problem Solving Method," IASTED Conference, Bozeman, Montana (USA), Aug. 1986.

[10] T. Sakaguchi and K. Matsumoto, "Development of a Knowledge Based System for Power System Restoration," IEEE Transactions on Power Apparatus and Systems, Vol. PAS-102, No. 2, Feb. 1983, pp. 320-329.

[11] R. Fujiwara, T. Sakaguchi, Y. Kohno, and H. Suzuki, "An Intelligent Load Flow Engine for Power System Planning," IEEE Transactions on Power Systems, Vol. PWRS-1, No. 3, Aug. 1986, pp. 302-307.

[12] C. C. Liu and K. L. Tomsovic, "An Expert System Assisting Decision-Making of Reactive Power/Voltage Control," IEEE Transactions on Power Systems, Vol. PWRS-1, No. 3, Aug. 1986, pp. 195-201.

[13] C. C. Liu, S. J. Lee and S. S. Venkata, "An Expert System Operational Aid for Restoration and Loss Reduction of Distribution Systems," accepted for 1987 PICA Conference, Montreal, Canada.

[14] F. Hein, "Expert System using Pattern Recognition by Real Time Signals," Proceedings CIGRE Conference, Paris, 1986.

[15] K. L. Tomsovic, C. C. Liu, P. Ackerman and S. Pope, "An Expert System as a Dispatchers' Aid for the Isolation of Line Section Faults," 1986 Transmission and Distribution Conference, Sept. 1986, and accepted for IEEE Transactions on Power Delivery.

[16] K. R. C. Mamandur, "Emergency Adjustments to VAR Control Variables to Alleviate Over-Voltages, Under-Voltages and Generator VAR Limit Violations," IEEE Transactions on Power Apparatus and Systems, Vol. PAS-101, No. 5, pp. 1040-1047, (May 1982).

[17] C. C. Liu, K. Tomsovic and S. Zhang, "Efficiency of Expert Systems as On-Line Operating Aids," 1987 PSCC Conference, Lisbon, Portugal.

[18] N. Wada, et. al., "A Real Time Expert System for Power System Fault Analysis," IASTED Conference, Bozeman, Montana(USA), (Aug. 1986).

[19] K. L. Tomsovic and C. C. Liu, "Bounding Computational Costs in Forward-Chaining Rule-Based Systems," in preparation.

[20] W. B. Gevarter, An Overview of Expert Systems, NBSIR 82-2505, National Bureau of Standards, Washington D.C., May 1982.

[21] C. Forgy, OPS83 Users Manual, Computer Science Department, Carnegie-Mellon University, 1983.

[22] M. Damborg and S. S. Venkata, Specification of Computer Aided Design of Transmission Protection Systems, Final Report EL-3337, RP 1764-6, EPRI, Jan. 1984.

[23] C. J. Date, An Introduction to Database Systems, Vol. 1, 3rd Edition, Addison-Wesley, 1982.

[24] J. Zaborszky, G. Huang and K. W. Lu, "A Textured Model for Computationally Efficient Reactive Power Control and Management," Paper No. 845SM 517-7, IEEE-PES Summer Meeting, July 1984.

[25] T. E. DyLiacco, "Real-Time Computer Control of Power Systems," Proceedings IEEE, Vol. 62, July 1974, pp. 884-891.

[26] F. F. Wu and C. C. Liu, "Characterization of Power System Small Disturbance Stability with Models Incorporating Voltage Variation," IEEE Transactions on Circuits and Systems, Vol. CAS-33, No. 4, April 1986, pp. 406-417.

[27] B. Stott and O. Alsac, "Optimized Load Flow with Steady-State Security," Paper T73 484-3, IEEE PES Summer Meeting, Feb. 1973, pp. 745-751.

[28] L. Brownston, et. al., Programming Expert Systems in OPS5, Addison-Wesley, New York, 1985, pp. 225-270.

[29] A. Gupta and C. Forgy, "Measurements on Production Systems," Technical Report CMU-CS-83-167, Carnegie-Mellon University, Dec. 1983.

[30] C. Forgy, "RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem," Artificial Intelligence, Vol. 19, No. 1, 1982, pp. 17-37.

[31] M. I. Schor, T. P. Daly, H. S. Lee and B. R. Tibbitts, "Advances in RETE Pattern Matching," Proceedings of AAAI, August 1986, pp. 226-232.

[32] C. Forgy, A. Gupta, A. Newell, R. Wedig, "Initial Assessment of Architectures for Production Systems," Proceedings of the NCAI, 1984, Austin, Texas, pp. 116-120.

[33] S. Stolfo and D. Miranker, "DADO: A Parallel Processor for Expert Systems," Proceedings 1984 International Conference on Parallel Processing, IEEE Computer Society Press, August 1984, pp. 74-82.

[34] G. E. Blelloch, "CIS: A Massively Concurrent Rule-Based System," Proceedings of AAAI, August 1986, pp. 735-741.

[35] R. J. Douglas, "Characterizing the Parallelism in Rule-Based Expert Systems," Proceedings of the Eighteenth Annual Hawaii International Conference on System Sciences, 1985, pp. 47-54.

VITA

PERSONAL

Birthdate: August 7, 1960
Birthplace: Slidell, Louisiana

EDUCATION

University of Washington 1982- 1987

Ph.D. Electrical Engineering Spring 1987
Advisor: C.C. Liu

M.S.E.E. 1984
Advisor: E. Noges
Thesis: "Analysis of a Non-Gyroscopic Inertial Measurement Unit"

Michigan Technological University 1978-1982

B.S.E.E. 1982
Emphasis: Control theory and computer engineering

HONORS

Eldec Career Development Fellowship, 1983-1986
Michigan Tech. Univ. Fund Scholarship, 1981-1982
Michigan Tech. Univ. Board of Control Scholarship, 1978-1982
Michigan Higher Education Association Scholarship, 1978-1982

EXPERIENCE

Academic

University of Washington: Research Assistant 1985-present
Responsibilities: Development of expert systems for on-line power system operational aid (EPRI project), preparation of proposals.

University of Washington: Teaching Assistant 1982-1985
Responsibilities: Lectured and graded courses/labs in the areas of computers, systems and controls, and circuits.

Michigan Tech.: Laboratory/Teaching Assistant 1981-1982
Responsibilities: Graded courses/labs in the areas of computers and systems.

Industrial

IBM: Summer engineering position 1983
Responsibilities: Development of PC based process controller for tape head plating bath.

IBM: Summer engineering position 1982
Responsibilities: Development of digital interface card for testing IBM system 38 CPU.

McDonnell Douglas Co.: Summer engineering position 1981
Responsibilities: Analog circuit design of aircraft actuator simulators, assistance to engineers in testing hydraulics system of short take off and landing aircraft.

PROFESSIONAL SOCIETIES

IEEE 1981-present
Circuit and Systems Society
Tau Beta Pi (Engineering Honor Society)

PUBLICATIONS

[1] "Reactive Power Compensation of Power Systems- An Expert System Approach", Proceedings of the 1984 IEEE International Conference on Systems, Man and Cybernetics,Oct. 1984, pp.85-89, (with C.C. Liu).

[2] "An Expert System Assisting Decision-Making of Reactive Power/Voltage Control", IEEE Transactions on Power Systems, Aug. 1986, pp. 195-201, (with C.C. Liu).

[3] "An Expert System as a Dispatchers' Aid for the Isolation of Line Section Faults," 1986 Transmission and Distribution Conference, Sept. 1986, and accepted for IEEE Transactions on Power Delivery, (with C.C. Liu, P. Ackerman and S. Pope).

[4] "Efficiency of Expert Systems as On-Line Operating Aids," Proceedings of the 1987 PSCC Conference, Lisbon, Portugal. (with C.C. Liu and S. Zhang).

[5] "Automating Performance Evaluation for a Reactive Power/Voltage Control Expert System," to be submitted to 1987 IEEE PES Winter Meeting and IEEE Transactions on Power Systems, (with C.C. Liu).

[6] "Bounding Computational Costs in Rule-Based Systems," in preparation, (with C.C. Liu).